University of Alberta

A Study of Match Cost Functions and Colour Use In Global Stereopsis

by

Daniel David Neilson

A thesis submitted to the Faculty of Graduate Studies and Research in
partial fulfillment of the requirements for the degree of

Doctor of Philosophy

Department of Computing Science

Fall 2009
Edmonton, Alberta

Examining Committee

Yee-Hong Yang, Computing Science

Walter Bischof, Computing Science

Nilanjan Ray, Computing Science

Vicky Zhao, Electrical and Computer Engineering

Frank Ferrie, Center for Intelligent Machines, McGill University

To my family, and my mother in particular. I would have never made it this far without your support and encouragement over the years.

Abstract

Stereopsis is the process of inferring the distance to objects from two or more images. It has applications in areas such as: novel-view rendering, motion capture, autonomous navigation, and topographical mapping from remote sensing data. Although it sounds simple, in light of the effortlessness with which we are able to perform the task with our own eyes, a number of factors that make it quite challenging become apparent once one begins delving into computational methods of solving it. For example, occlusions that block part of the scene from being seen in one of the images, and changes in the appearance of objects between the two images due to: sensor noise, view dependent effects, and/or differences in the lighting/camera conditions between the two images.

Global stereopsis algorithms aim to solve this problem by making assumptions about the smoothness of the depth of surfaces in the scene, and formulating stereopsis as an optimization problem. As part of their formulation, these algorithms include a function that measures the similarity between pixels in different images to detect possible correspondences. Which of these match cost functions work better, when, and why is not well understood. Furthermore, in areas of computer vision such as segmentation, face detection, edge detection, texture analysis and classification, and optical flow, it is not uncommon to use colour spaces other than the well known RGB space to improve the accuracy of algorithms. However, the use of colour spaces other than RGB is quite rare in stereopsis research.

In this dissertation we present results from two, first of their kind, large scale studies on global stereopsis algorithms. In the first we compare the relative performance of a structured set of match cost cost functions in five different global stereopsis frameworks in such a way that we are able to infer some general rules to guide the choice of which match cost functions to use in these algorithms. In the second we investigate how much accuracy can be gained by simply changing the colour representation used in the input to global stereopsis algorithms.

Acknowledgements

There are so many people without whom I would have never made it to this point. I could probably write another document that is at least the length of this one were I to thank them all individually.

I will start with my family, without whom I would not be the person that I am today. I would like to express my undying gratitude to my mother. There is no doubt in my mind that I would never have accomplished even half the things that I have during my life without her constant encouragement and support. I would also like to thank my grandparents, aunts, and uncle for just being there. To my brother Ryan, thank you for being the ying to my yang; everyone needs their opposite to help put things in perspective, and you definitely fit the bill for me. Last but certainly not least, thank you to my wife Christine; I would have gone insane long ago if not for your love and patience.

Thank you so very much to Dr. Herb Yang for being my supervisor for the past eight years. I am indebted to you for giving me just enough rope to explore my own interests while not hanging myself in the process. I cannot imagine having done my graduate studies any other way. I hope that I do you proud in my professional endeavors, and look forward to continuing to collaborate with you in the future.

Thank you to my high school math teacher, William Korytowski, for nurturing that spark of interest I had in mathematics; that interest has been guiding me ever since. Thank you to Dr. Neil Arnason, and Dr. William Kocay at the University of Manitoba for giving me a chance at academic research for all of those summers during my undergraduate program. I would have never even considered graduate school without those opportunities.

Finally, to all of my friends and fellow graduate students: thank you for the distractions. The injections of fun have always been, and will always be, a welcome distraction – especially when I just need to step away from the work for a while. Grad school is about more than just the work, and you were always there to remind me of that fact.

Contents

List of Figures

List of Tables

List of Notation

s, α	– Scalar value.
	–
$\mathbf{p}$	– n-dimensional point.
$\vec{\mathbf{v}}$	– n-dimensional vector.
$\mathbf{p}_i$,$\vec{\mathbf{v}}_i$	– The i^{th} element of point $\mathbf{p}$ or vector $\vec{\mathbf{v}}$
	–
$\mathbf{M}$	– An $n \times m$ matrix.
$\mathbf{M}_{i,j}$	– The element of $\mathbf{M}$ at row i and column j.
	–
$\mathcal{I}$	– An image.
$\mathbf{p} \in \mathcal{I}$	– $\mathbf{p}$ is a pixel in $\mathcal{I}$.
$d(\mathbf{p})$	– Depth at pixel $\mathbf{p}$.
$\delta(\mathbf{p})$	– Disparity at pixel $\mathbf{p}$.
$\vec{\mathbf{c}}(\mathbf{p})$	– Colour vector of pixel $\mathbf{p}$.
	–
$\rho(X)$	– The marginal/prior probability of X.
$\rho(X\|Y)$	– The conditional probability of X, given Y.
$\rho(X, Y)$	– The joint probability of X and Y.

Chapter 1

Introduction

Stereopsis is the problem of inferring depth (distance from the camera) of objects in a scene from a binocular pair of stereo images. Early stereopsis algorithms focused mostly on identifying features in the input images, such as object/intensity edges or corners, and identifying which features are the same in both images. Modern stereopsis algorithms, on the other hand, aim to solve this problem by identifying correspondences for even non-feature pixels in the input images. The displacement between the correspondences, along with the relative position of the two cameras, is then used to calculate depth.

Although this problem may sound simple, in light of the effortlessness with which we are able to perform the task with our own eyes, a number of factors that make it quite challenging become apparent once one begins delving into computational methods of solving it. Occlusions that block part of one image from being seen in the other prevent correspondences of the occluded pixels from being found at all; forcing stereopsis algorithms to rely on other mechanisms, such as smoothness assumptions, to assign a depth value to these pixels. Additionally, the appearance of a point in the imaged scene can have a different appearance/colour in each of the input images; making matching solely based off of colour potentially unreliable. Camera sensor noise, view-dependent effects (mirrors, non-Lambertian objects, refraction, et cetera), subtle differences in colour due to natural anti-aliasing from the colour interpolation that is performed in the camera-sensor, and global colour intensity changes caused by differences in camera or lighting conditions used to capture each input image can all cause differences in the appearance of scene points as viewed in each image.

Part of what makes the computational stereopsis problem interesting are its applications. For instance, novel-view synthesis is the task of taking a set of images of a scene that are captured with an array or grid of cameras, and generating a new view of the scene that is not present in any of the input. Given depth maps of all of the input images, this task can be solved by any number of rendering algorithms. A geometric model for an object is typically created using laser range scanners, but optical techniques based on computational stereopsis can be applied in place of these scanners. Motion capture, in its simplest form, requires determining the correspondences between numerous tracking tags; techniques for finding feature correspondences in computational stereopsis are applied to help solve this problem. Recently, computational stereopsis algorithms [12] have even been applied to create motion capture data without using marker tags on the subjects being captured. To avoid colliding with objects, autonomous navigation requires knowledge of the robot's surroundings [28]. Such knowledge can be acquired with laser range scanners, radar, or computational stereopsis techniques; see [28] for more details. Topographical mapping can also be accomplished by applying computational stereopsis techniques to remote sensing data [45].

Stereopsis algorithms can be broadly categorized into two categories [82]: local and global algorithms. Local algorithms attempt to infer depth at each pixel in an input image using only information about pixel correspondences local to the pixel. Global algorithms, on the other hand, make an explicit assumption about the smoothness of the depth in an image, and infer depth by solving an optimization problem.

All algorithms that seek to solve the stereopsis problem rely on a *match cost function*, that measures the similarity between two pixels. This functions helps determine the likelihood that the pixels are projections of the same three dimensional point. Which match cost functions work better, why, and under what circumstances is not well understood. Additionally, when storing an image in a computer, the colours in an image can be stored using any of a number of different colour representations. For example, RGB represents a colour using the quantity of the red, green, and blue primaries that must be mixed to obtain the colour. Although the RGB representation is the most common representation used by stereopsis algorithm researchers, some research [13, 48, 61] indicates that algorithms can yield more accurate depth estimates when a different colour representation is used.

In this dissertation we conduct two, first of their kind, large scale studies aimed at furthering our understanding of the components of global stereopsis algorithms. In the first study we perform a systematic study on the relative performance of several hundred match cost functions, in the context of five different global stereopsis frameworks, on a large data set. The match cost functions that we use are built from component parts in such a way that we are able to make observations about the relative performance of those parts by analyzing the relative performance of the match cost functions that use them. In the second study we conduct a large scale comparison of the effect on global stereopsis algorithm accuracy obtained by changing only the colour representation used in the input images to the algorithm. We compare the relative accuracy changes observed by using 30 colour representations of a large data set consisting of synthetic and photographic image pairs for input to 75 global stereopsis algorithms formed by combining five global stereopsis frameworks and 15 match cost functions

This dissertation is structured as follows. In Chapter 2 we describe the stereopsis problem, and existing methods for solving it. In Chapter 3 we discuss models of colour vision, mathematical representations of colour, and some of the uses of colour representation in stereopsis research. In Chapter 4 we present our methodology for evaluating stereopsis algorithms. Chapter 5 presents the results of our large scale study on the relative performance of match cost functions within global stereopsis algorithms. We present the results of our comparison of colour representations for input into global stereopsis algorithms in Chapter 6, and conclude in Chapter 7.

1.1 Contributions

One potential pitfall with existing frameworks for evaluating stereopsis algorithms is that the choice of user-parameters may impact the results of the evaluation unless the researcher spends a lot of time manually tweaking user-parameters for the algorithms being compared. Sometimes these user-parameters are input sensitive and should be changed for every image pair in the researcher's data set to obtain optimal (or near-optimal, at the least) results. If care is not taken to ensure that the user-parameters used in the evaluation are as close to optimal as possible then the results of any comparative evaluation may reflect the poor parameter choices rather than any real properties of the algorithms being compared.

A common, and disturbing, trend among computational stereopsis researchers is that there is no effort made to determine whether there is any statistical significance to the results of the comparisons made. It is quite possible, and even likely, that there have been comparisons made between methods that conclude that some proposed method is superior when, in fact, there is actually no statistically significant difference between them.

In this dissertation we develop a novel framework for evaluating the relative performance of stereopsis algorithms. To ensure that the evaluation is being performed on a level playing field, our framework incorporates a response-surface method [65] for optimizing user-parameters in the stereopsis algorithms being compared. Response-surface methods build a model of an objective function, f_{obj}, using all available data about the objective, select a new point at which to evaluate f_{obj}, evaluate f_{obj} at the selected point, and incorporate the new data into the model. This process is then repeated as many times as desired. Large scale studies of stereopsis algorithms require automated parameter optimization methods such as this due to the input-sensitivity of some user-parameters. As part of this evaluation framework we also propose a ranking algorithm that will assign the same rank to algorithms that produce results that cannot be statistically differentiated at a given confidence level. We apply our evaluation framework to both of our studies.

This dissertation contains results from the first ever systematic large scale study on the relative performance of match cost functions for global stereopsis algorithms. From the results of this study we are able to infer a set of general guidelines for developing high quality match cost functions for use in global stereopsis algorithms. This study also provides ranking data that is statistically significant at 95% confidence for over one hundred match cost functions, and their constituent parts, within the context of global stereopsis algorithms. This information can be used by researchers and developers to help guide their choice on which match cost function, or functions, to consider for use in their own stereopsis algorithms.

We also present results from the first large scale comparison of the effect on global stereopsis algorithm accuracy obtained by only changing the colour representation used for input images. This study shows that chaging the input images from RGB to a perceptually uniform colour space, such as CIELuv and CIELab, can improve the accuracy of global stereopsis algorithms; we observe a median improvement in accuracy of as much as 7.2% by simply changing the input to one of these colour spaces. We also show that some of the depth-smoothing assumptions made in global stereopsis algorithms are more amenable to accuracy improvements from changing the colour space of the input images to a perceptually uniform colour space than other smoothing assumptions. Finally, we include much of the raw data on the relative improvement in accuracy obtained by changing the input images from the RGB colour space to each of the 29 other colour spaces in our study; this data can be used by researchers and developers in their own work.

Chapter 2

Stereopsis

Stereopsis refers to the process of inferring depth from binocular visual sensory input. Given a binocular pair of stereo images, computational stereopsis algorithms aim to solve the problem of calculating depth, or the equivalent quantity disparity, of the objects visible in the scene captured by the input.

There are many factors that make this problem particularly difficult, and interesting. According to Šára [95], the difficulty in the stereopsis problem arises from three independent factors:

1. *Data uncertainty due to insufficient signal-to-noise ratio in images of weakly textured objects;*

2. *Structural ambiguity due to the presence of periodic structures combined with the inability of a small number of cameras to capture the entire light field; and*

3. *Task formulation: If the stereoscopic vision goal is posed as a scene reconstruction from image projections the problem is that the same image set can be generated by infinitely many scenes.*

We would also add that since the images are typically not synthetic, the camera sensor will introduce some noise to the colour recorded for each pixel. Furthermore, the presence of view dependent shading effects such as non-Lambertian *bi-directional reflectance functions* (BRDFs) in the scene, and natural anti-aliasing caused by interpolation that takes place in the camera sensor can cause difficulty. These effects will cause some corresponding pixels to have drastically different colours in different images; making matching based solely off of pixel colour, or even colour in a small neighbourhood of the pixel, inaccurate. Additionally, occlusion of scene components due to differing viewpoints causes some pixels in an image to have no corresponding pixel in a second image. Assigning depth to such pixels must rely on mechanisms other than the match cost function; such as the smoothness assumptions made by global stereopsis algorithms.

Many different methods for solving the stereopsis problem have been proposed over the past couple of decades. These methods can be broadly categorized into two categories: local and global algorithms. Local algorithms find pixel correspondences using only information local to each pixel, whereas global algorithms assume that depth follows certain smoothness constraints and solves an optimization problem. In this Chapter, we discuss the computational stereopsis problem, and methods for solving it.

2.1 The Computational Stereopsis Problem

Given a pair of images, $\mathcal{I}_s$ and $\mathcal{I}_m$, depicting a three dimensional scene from two viewpoints. The goal of a computational stereopsis algorithm is to infer the distance from the camera to the three dimensional objects visible in those images. However, rather than calculating the depth, $d(\mathbf{p})$, for each pixel, $\mathbf{p}$, in the given images, computational stereopsis algorithms typically calculate disparity, $\delta(\mathbf{p})$, at each pixel.

Disparity is a quantity proportional to inverse depth, and has two interchangable definitions: one unitless, and one in pixel units. Let s be the distance between the camera origin and the three dimensional viewplane position of pixel $\mathbf{p}$; see Figure 2.1. Then, the unitless disparity, $\delta'(\mathbf{p})$, of $\mathbf{p}$ can be calculated from its depth as:

$$\delta'(\mathbf{p}) = \frac{s}{d(\mathbf{p})}. \tag{2.1}$$

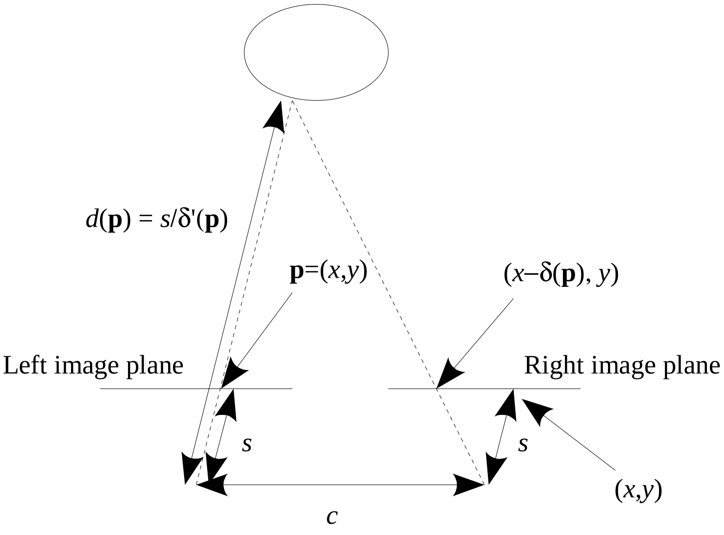


Figure 2.1: The relationship between depth, unitless disparity, and pixel-unit disparity.

The pixel units quantity for disparity, $\delta(\mathbf{p})$, gives the image space displacement between the projections of the corresponding three dimensional point to both images in the image pair. It can be calculated as:

$$\delta(\mathbf{p}) = c\frac{s}{d(\mathbf{p})}. \tag{2.2}$$

where c is the distance between the two cameras in pixel units:

Assuming that $\mathcal{I}_s$ and $\mathcal{I}_m$ are rectified (the warpings of a pixel at every depth value lies on the same scanline in the other image), the pixel corresponding to $\mathbf{p} = (x, y)$ is at location $(x + \delta(\mathbf{p}), y)$ or $(x - \delta(\mathbf{p}), y)$ depending on whether $\mathbf{p}$ is in the right or left image of the image pair, respectively. Throughout this dissertation we use this definition of disparity.

For a pixel, $\mathbf{p}$, of a given image we try to determine the depth of the point on the object, visible at $\mathbf{p}$ in a manner inspired by the human visual system. That is, if we can find the corresponding point, if any, in the other given image we can triangulate the three dimensional position of the point by using the known camera information (relative positions and orientations) for the images, and the positions of $\mathbf{p}$ and its corresponding pixel on the two image planes. However, finding these corresponding points can prove to be very difficult, and, thus, has been the focus of intense research over the last 20 years.

In their taxonomy of dense stereopsis algorithms, Scharstein and Szeliski [82] present a four phase decomposition of stereopsis algorithms:

1. Matching cost computation;

2. Cost (support) aggregation;

3. Disparity computation / optimization; and

4. Disparity refinement.

If we are calculating a disparity map for some image, $\mathcal{I}_s$, then the match cost computation step calculates a *disparity space image* (DSI) for $\mathcal{I}_s$. This DSI is simply an image with the same dimensions as $\mathcal{I}_s$ where each pixel, $\mathbf{p}$, stores the values of the match cost function, $c(\mathbf{p}, \delta)$, for every possible disparity value, δ, rather than storing colour information.

In phase two, cost aggregation is used by some stereopsis algorithms to "aggregate the matching cost by summing or averaging over a *support region* in the DSI" [82]. Basically, what this means is that each value in the DSI is convolved with a two or three dimensional support function. Alternatively, as in Scharstein and Szeliski [81], the cost aggregation step can be performed using an iterated diffusion operation.

In the disparity computation/optimization phase, an initial (or final if phase four is omitted) disparity map is generated by the information available in the supplied DSI. Most algorithms can be classified as either *local* or *global* algorithms. Local algorithms tend to rely on the work done in the match cost computation and cost aggregation phases being sufficiently accurate to determine the disparity at each pixel; so, they tend to choose disparity values through a simple local selection method like winner-take-all. Global methods, on the other hand, perform most of their work in phase three. These global methods make assumptions about the smoothness of the disparity map, and solve an optimization problem to calculate the disparity map.

The disparity refinement phase is used by some algorithms to remove some of the quantization present in the disparity map that is the result of using a small number of quantized disparity values. Quantization can be removed by fitting a surface, such as a Bezier patch, to the disparity map and then reassigning the values for the disparity map from the surface. Alternatively, when calculating a disparity map for multiple images Dainese et al. [24] propose creating a three dimensional level set from all of the disparity maps calculated, which can then be used to generate the final disparity maps by projecting the level set surface back into each image. This phase can also be used to reduce the number of "salt and pepper" outliers present in the disparity map using a median filter.

2.2 Match Cost Functions

The main, and probably the most important, component of every stereopsis algorithm is the match cost function. This function measures the fitness of assigning a disparity value, δ, to a pixel $\mathbf{p}$. This fitness is evaluated by warping $\mathbf{p}$ at disparity δ into the other given image and comparing the information (such as colour, or colour gradient) at $\mathbf{p}$ with the information at the warped-to pixel.

Unless otherwise specified, throughout this dissertation we assume that all of our match cost functions are used to measure the matching cost using RGB colour pixels. In all cases, the match cost functions that we use can be trivially modified to use greyscale pixels, pixels in other colour spaces, pixels from a gradient or Laplacian image, or other pixel types.

There are many ways to define a match cost function. One can use a simple Euclidean distance metric to slightly more complex spatially aggregated matching functions that compare the regions of colour around the pixel under consideration and its warp to the other image in the binocular pair. In this section we present the match cost functions investigated in this dissertation.

2.2.1 The Birchfield and Tomasi Measure

Birchfield and Tomasi [6] propose a match cost function that has become very popular among computational stereopsis researchers. This match cost function, hereafter referred to as the *BT* measure, is provably insensitive to errors that can arise from image sampling. This is achieved by comparing two pixels using the linearly interpolated intensity functions surrounding them.

Let $\mathbf{p}$ and $\mathbf{q}$ be pixels from different images such that $\mathbf{q}$ is the warp of $\mathbf{p}$ at some disparity value, δ. As defined, the BT measure requires that the epipolar line of $\mathbf{p}$ be the horizontal scanline containing $\mathbf{q}$, and vice versa; this is because the left and right neighbours of $\mathbf{p}$ and $\mathbf{q}$ are used to linearly interpolate the intensity curves on the epipolar lines. Furthermore, the BT measure is only defined for greyscale image pixels. In practice, when the match cost between colour pixels is desired, the BT measure is used on each colour channel separately (e.g. [79]) and the results are summed, or squared and summed, to yield a colour match cost function.

Let $\mathbf{p^L}$ and $\mathbf{p^R}$ denote the pixels to the left and right of $\mathbf{p}$, respectively. Also, let $\vec{\mathbf{c}}(\mathbf{p})$ denote the colour of the pixel $\mathbf{p}$. Then, the BT measure is defined as:

$$c^{BT}(\mathbf{p},\delta) = \min\{\bar{d}(\mathbf{p},\mathbf{q}), \bar{d}(\mathbf{q},\mathbf{p})\} \tag{2.3}$$

where $\bar{d}(\mathbf{p},\mathbf{q})$ is the lowest match cost between $\mathbf{p}$ and the pixels on the line $\mathbf{q^L}$-$\mathbf{q^R}$:

$$\begin{aligned}
\bar{d}(\mathbf{p},\mathbf{q}) &= \min_{0\leq t\leq 1} |\vec{\mathbf{c}}(\mathbf{p}) - \vec{\mathbf{c}}(\mathbf{q^L}t + (\mathbf{q^R} - \mathbf{q^L})(1-t))| && (2.4a)\\
&= \max\{0, \vec{\mathbf{c}}(\mathbf{p}) - \max\{\vec{\mathbf{c}}(\mathbf{q^L}), \vec{\mathbf{c}}(\mathbf{q}), \vec{\mathbf{c}}(\mathbf{q^R})\}, \min\{\vec{\mathbf{c}}(\mathbf{q^L}), \vec{\mathbf{c}}(\mathbf{q}), \vec{\mathbf{c}}(\mathbf{q^R})\} - \vec{\mathbf{c}}(\mathbf{p})\}. && (2.4b)
\end{aligned}$$

2.2.2 Constructable Match Cost Functions

Many of the match cost functions in use today can be constructed from up to four base components: a *channel function* (CF), *channel norm function* (CNF), *channel aggregate* (CA), and an optional *spatial aggregate* (SA). The first two components (CF and CNF) together define greyscale, or single-channel, match cost functions. To create a match cost function for colour pixels, a greyscale measure is then applied to each colour channel separately and combined via a channel aggregate. The first three components can be applied to all pixels in an image at all desired disparity values to construct a DSI. Once the DSI for an image has been constructed, a *spatial aggregate* can be applied to the values in the DSI to construct a match cost function that takes into account a spatial neighbourhood around each pixel.

For example, the sum of squared differences (SSD) match cost function uses a difference operator for its CF, a squaring function for its CNF, and a summation for its CA to yield the function:

$$c_{SSD}(\mathbf{p},\delta) = \sum_i (\vec{\mathbf{c}}(\mathbf{p})_i - \vec{\mathbf{c}}(\mathbf{q})_i)^2 \tag{2.5}$$

where $\mathbf{q}$ is the warp of $\mathbf{p}$ at disparity δ, and i runs over each colour channel in the colour representation being used.

Function Name	**Abbreviation**	**Function**
Difference	D	$g(a, b) = \lvert a - b \rvert$
Birchfield & Tomasi	B	See Section 2.2.1

(a) Channel functions.

Function Name	**Abbreviation**	**Function**
L1 norm	A	$f(x) = \lvert x \rvert$
L2 norm	S	$f(x) = x^2$
Generalized Exponential	$\mathrm{G}_{\sigma,s}$	$f_{\sigma,s}(x) = \lvert x/\sigma \rvert^s$
Lorentzian p-function [7]	L_{σ}	$f_{\sigma}(x) = \ln(1 + \frac{1}{2}(\frac{x}{\sigma})^2)$

(b) Channel norm functions.

Aggregate Name	**Abbreviation**	**Aggregate**
Sum	S	$c(\mathbf{p}, \delta) = \sum_i f(x_i)$
Square root	Sq	$c(\mathbf{p}, \delta) = \sqrt{\sum_i f(x_i)}$
Weighted sum	$\mathrm{Sw}_{w_1,\ldots,w_n}$	$c(\mathbf{p}, \delta) = \sum_i w_i f(x_i)$
Max	Mx	$c(\mathbf{p}, \delta) = \max\{f(x_i) : i\}$
Sum minus max	Sx	$c(\mathbf{p}, \delta) = [\sum_i f(x_i)] - \max\{f(x_i) : i\}$

(c) Channel aggregates.

Aggregate Name	**Abbreviation**	**Weight Function**
Mean	M	$w_m(\mathbf{p}, \vec{\mathbf{x}}) = 1$
Spatial weight	Ws	$w_{Ws}(\mathbf{p}, \vec{\mathbf{x}}) = e^{-\frac{\lvert\lvert\vec{\mathbf{x}}\rvert\rvert}{\alpha}}$
Colour weight	Wc	$w_{Wc}(\mathbf{p}, \vec{\mathbf{x}}) = e^{-\frac{\lvert\lvert\vec{\mathbf{c}}(\mathbf{p})-\vec{\mathbf{c}}(\mathbf{p}+\vec{\mathbf{x}})\rvert\rvert}{\beta}}$
Bilateral	B	$w_B(\mathbf{p}, \vec{\mathbf{x}}) = w_{Ws}(\mathbf{p}, \vec{\mathbf{x}}) w_{Wc}(\mathbf{p}, \vec{\mathbf{x}})$
Adaptive support weight	Wa	$w_{Wa}(\mathbf{p}, \vec{\mathbf{x}}) = w_B(\mathbf{p}, \vec{\mathbf{x}}) w_B(\mathbf{q}^{\mathbf{p},\delta}, \vec{\mathbf{x}})$

(d) Spatial aggregate weighting functions.

Table 2.1: The channel functions, channel norm functions, channel aggregates, and spatial aggregates used in match cost functions within this dissertation.

Table 2.1 summarizes the match cost function components used in this dissertation. In addition to the channel norm functions and channel functions listed in Tables 2.1b and 2.1c, respectively, we also include truncated – taking the min of the function and a user-supplied constant – versions of them all except for the *Max* channel aggregate. We use the abbreviations included for each component in these tables for easy specification of a particular match cost function – abbreviations are listed in the order: CA, CNF, CF. Furthermore, we post-fix the channel aggregate and channel norm function symbols with "[t]" to indicate that particular component is to be truncated. For example, instead of writing $\sum_i \min\{\tau, \lvert\frac{\vec{\mathbf{c}}(\mathbf{p})_i - \vec{\mathbf{c}}(\mathbf{q})_i}{\sigma}\rvert^s\}$ to indicate the match cost function that uses the difference operator as its channel function, truncated generalized exponential as its channel norm function, and a summation for a channel aggregate we would instead simply write "$\mathrm{SG}_{\sigma,s}$[t]D" – in practice, we occasionally drop the parameters for each component from the abbreviation and write "SG[t]D."

A potential problem with match cost functions that do not use a spatial aggregate is that they are insensitive to the context of the pixel being evaluated. That is, given a pixel $\mathbf{p}$ such that there exist disparity values $\delta_1 \neq \delta_2$ for which the warp of $\mathbf{p}$ at δ_1 and δ_2 are pixels with the same colour then the match costs of $\mathbf{p}$ at δ_1 and δ_2 will be identical. To get around this, many researchers incorporate a spatial aggregate (also referred to as a template or window) that uses information from all pixels in a small $(2n + 1) \times (2m + 1)$ rectangle centered on $\mathbf{p}$ and the warp of $\mathbf{p}$.

On top of the components listed in Tables 2.1a, 2.1b, and 2.1c we also use five different spatial aggregates in this thesis. These five spatial aggregates all result in match cost functions with the form:

$$c(\mathbf{p},\delta) = \frac{\sum_{\vec{\mathbf{x}}\in\mathcal{R}} w(\mathbf{p},\vec{\mathbf{x}})c'(\mathbf{p}+\vec{\mathbf{x}},\delta)}{\sum_{\vec{\mathbf{x}}\in\mathcal{R}} w(\mathbf{p},\vec{\mathbf{x}})} \tag{2.6}$$

where $c'(\mathbf{p},\delta)$ denotes a match cost function, $\mathcal{R} = \{(x,y) : x \in [-n,n], y \in [-m,m]\}$, $w(\mathbf{p},\vec{\mathbf{x}})$ denotes one of the weighting functions in Table 2.1d, α and β are user-parameters, and $\mathbf{q}^{\mathbf{p},\delta}$ denotes the pixel obtained by warping pixel $\mathbf{p}$ at disparity δ.

2.3 Local Stereo Algorithms

There are literally hundreds, if not thousands, of different local stereo correspondence algorithms. In this section we will present a fairly small sampling of the algorithms that have been proposed over the years.

2.3.1 The Winner-Take-All Algorithm

The winner-take-all (WTA) algorithm is the simplest disparity computation algorithm. Given the values of a match cost function, $c(\mathbf{p},\delta)$, for image $\mathcal{I}$ the WTA algorithm finds the disparity map, f, for $\mathcal{I}$ as:

$$f(\mathbf{p}) = \arg\min_{\delta} c(\mathbf{p},\delta) \tag{2.7}$$

for all pixels, $\mathbf{p}$, in $\mathcal{I}$.

The main strength of the WTA algorithm is its speed; even for large images with a lot of potential disparity values, it is extremely fast to calculate the disparity map f. However, the cost of this speed is a heavy susceptibility to the quirks of the match cost function. For instance, if for some pixel $\mathbf{p}$ there are n disparity values, $\delta_1, \delta_2, \ldots, \delta_n$, for which $c(\mathbf{p},\delta_i) = c(\mathbf{p},\delta_j)$ for all $i,j \in [1,n]$ (due to a region of uniform colour, for example), then the choice of $f(\mathbf{p})$ made by the WTA algorithm is effectively random. Furthermore, in this case the choice of $f(\mathbf{p})$ will not necessarily even be similar to $f(\mathbf{q})$ for the neighbouring pixels, $\mathbf{q}$, of $\mathbf{p}$; which would clearly be a problem if the object visible through $\mathbf{p}$ is something flat, like a wall.

2.3.2 Other Algorithms

The WTA algorithm is a common base component in a number of local stereo algorithms. For instance, Chen and Medioni [19], Zhang et al. [109], Wei and Ngan [96], and Goesele et al. [36] all present local stereo algorithms that use the WTA algorithm in some manner.

Chen and Medioni [19] propose a "volumetric stereo matching method" that calculates a dense disparity map for each image in an input set, $\mathbb{I}$, through a region growing method. To calculate the disparity map for $\mathcal{I}_i \in \mathbb{I}$, they begin by calculating a set of *seed pixels*, S_i, as the set of pixels which are assigned high confidence disparity assignments through a WTA algorithm; that is, the disparity assignments for the seed pixels correspond to a unique local minimum in the match cost that is below a user supplied threshold. The disparity for each pixel in S_i is also set to their WTA assigned disparity in this step. Once S_i has been constructed, all of the pixels in the set are placed into a priority queue with the highest quality (lowest match cost) given the highest priority. Letting Q be the priority queue, the region growing phase of the algorithm then proceeds as follows:

1. Pop the highest priority pixel, $\mathbf{p}$, off the top of Q.

2. For each four-connected neighbour, $\mathbf{p'}$, of $\mathbf{p}$ do:
    (a) Set $\delta = \arg\min_{\delta \in \{f_i(\mathbf{p})-1, f_i(\mathbf{p}), f_i(\mathbf{p})+1\}} c(\mathbf{p'}, \delta)$.
    (b) If $c(\mathbf{p'}, \delta) < c(\mathbf{p'}, f_i(\mathbf{p'}))$ then set $f_i(\mathbf{p'}) = \delta$ and push $\mathbf{p'}$ onto Q.
3. If Q is not empty, then goto (1).

where $c(\mathbf{p}, \delta)$ denotes the match cost function being used.

Zhang et al. [109] use the WTA algorithm in a multiresolution framework in an attempt to improve the running time of a stereopsis algorithm. Given a pair of images, $\mathcal{I}_s$ and $\mathcal{I}_m$, Zhang et al. begin by calculating a set of mipmaps for each image, $\{\mathcal{I}_s^0, \mathcal{I}_s^1, \ldots, \mathcal{I}_s^n\}$ and $\{\mathcal{I}_m^0, \mathcal{I}_m^1, \ldots, \mathcal{I}_m^n\}$, where $\mathcal{I}_j = \mathcal{I}_j^0$ and $\mathcal{I}_j^k$ is the k^{th} mipmap level of image $\mathcal{I}_j$; that is, if $\mathcal{I}_j$ is an $n \times n$ pixel image, $\mathcal{I}_j^k$ is the $\frac{n}{2^k} \times \frac{n}{2^k}$ image obtained by averaging $\mathcal{I}_j^{k-1}$. The term *mipmap* was introduced in Williams [102]; *mip* stands for the latin "multum in parvo" which means "many things in a small place." Then, to find the disparity of pixel $\mathbf{p}$ in, say, $\mathcal{I}_s$ the epipolar line of $\mathbf{p}$ in $\mathcal{I}_m^n$ is searched for the pixel, $\mathbf{q^n}$, with lowest match cost to $\mathbf{p}$. The next step is to iterate; given $\mathbf{q^k}$ in $\mathcal{I}_m^k$ that most corresponds with $\mathbf{p}$, $\mathbf{q^{k-1}}$ is found by searching the epipolar line of $\mathbf{p}$ in image $\mathcal{I}_m^{k-1}$ constrained to the four pixels that correspond to $\mathbf{q^k}$. When $\mathbf{q^0}$ is found, the iteration stops and the disparity, δ, that would map $\mathbf{p}$ to $\mathbf{q^0}$ is taken to be the disparity of $\mathbf{p}$.

The primary benefit of Zhang et al.'s algorithm is that it is faster than the stock WTA algorithm. However, this speed comes with a potential pitfall. Since the algorithm limits the search for the disparity of $\mathbf{p}$ to the portion of the epipolar line that intersects pixel $\mathbf{q}_k$ (for any k), it is possible that the search may be erroneously limited to the wrong portion of the image if, for example, noise causes a pixel in the wrong portion of $\mathcal{I}_m^k$ to be chosen as the most likely corresponding pixel to $\mathbf{p}$.

Wei and Ngan [96] use the WTA algorithm as the first step of their algorithm. The algorithm calculates disparity maps for binocular stereo images by first performing a WTA on edge pixels and keeping only those disparity assignments that have a match cost below a user supplied threshold and satisfy the *strong consistency constraint* [39] (i.e. if $\mathbf{p}$ warps to $\mathbf{q}$ under its disparity assignment, then $\mathbf{q}$ also warps to $\mathbf{p}$ under its disparity assignment). The second step of the algorithm propagates the disparity assignments along the edges in both images to result in disparity assignments for every edge pixel in both images where all disparity assignments are consistent. Finally, the non-edge pixels are assigned disparities by interpolating from the nearby assigned disparity values; the next pixel chosen for interpolation is always a pixel with the highest density of neighbour pixels with assigned disparity values.

Goesele et al. [36] also use the WTA algorithm in the first phase of their multiview stereo algorithm to calculate a sparse initial disparity map for each image that is then used in later phases. In the first phase of their algorithm, Goesele et al. use the WTA algorithm to calculate the best disparity assignment for each pixel in all input images; only the assignments which are below a user supplied threshold match cost are kept. In the second phase of the algorithm this sparse disparity assignment is used to create a signed distance field from which they extract a level set for the object depicted in the input images. This level set can then be projected back into each input image to result in a disparity assignment for every pixel.

Collins [22] proposes a multiview stereo algorithm for calculating disparity values for image features, such as object edges, that does not use the WTA algorithm in its formulation. This *sweep-space* approach requires that all cameras used to capture the input images are facing the same general direction. The algorithm works by sweeping a plane away from the image plane with a discrete step size. At each step, all of the pixels in all images that correspond to an image feature are projected to the current plane; if a voxel in the swept plane receives more than a statistically determined number of feature projections, then the center of the voxel is set as the 3D position of the feature.

Moallem and Faez [68] introduce the concept of the *directional derivative of disparity* (DDD) to limit the range of disparity values that must be considered when using the WTA algorithm to calculate the disparity of edge pixels. The DDD of an edge pixel, $\mathbf{p}'$, yields a range of disparity values within which the disparity of adjacent edge pixels is highly probable of occurring. So, when searching for the disparity of an edge pixel, $\mathbf{p}$, the DDD of adjacent edge pixels can be used to limit the search range. This ability to limit the search range for a disparity value is then used in an algorithm that begins by performing an edge detection on $\mathcal{I}$ to identify which pixels belong to object edges; these edge pixels are partitioned into sets of vertical lines. Processing begins in the upper-left corner of the image, and proceeds right then down. If an edge pixel, $\mathbf{p}$, is the first in its partition to be processed, the DDD cannot limit the search for the disparity for $\mathbf{p}$; on the other hand, if $\mathbf{p}$ is not the first pixel in its partition to be processed then the disparity of the pixel above $\mathbf{p}$ is used to limit the search range for disparities of $\mathbf{p}$.

Šára [95] proposes a sparse stereopsis algorithm based on the idea of inhibition zones. This algorithm begins with the set, S, of all possible disparity assignments for every pixel in both input images. This set is then processed in the order from the highest quality matches to the lowest quality; when a match is processed, it is added to a second set of "pseudo-stable" matches, P, and the matches in S that conflict with the match are removed from S. The set P is then processed to qualify the matches in P as either stable (which make up the final disparity assignment output by the algorithm) or inhibited.

2.4 Global Stereopsis Algorithms

Global stereopsis algorithms seek to solve the stereopsis problem through a global formulation, defined over the disparity function that is solved via an optimization technique. The global formulation can take the form of an energy equation over the disparity map with lower energy values corresponding to higher quality disparity assignments, or a probability formulation that measures the probability of a particular disparity map being correct.

In this section we present global formulations for the computational stereopsis problem that use both a probability formulation and an energy formulation. We then present a number of different methods that have been used in the research community to find approximately optimum solutions for these formulations.

2.4.1 Global Formulations of Stereopsis

Most global stereopsis formulations being used today either use or are derived from a Bayesian probability formulation. In this section we present the Bayesian probability formulation of the stereopsis problem and show how this formulation is morphed into an energy minimization formulation.

Probabilistic Formulations

In stereopsis we are given input images, $\mathcal{I}_s$ and $\mathcal{I}_m$, and it is our desire to calculate a disparity map, f_s, for $\mathcal{I}_s$. In the probabilistic framework, this is expressed as a solution that maximizes the posterior probability $\rho(f_s|\mathcal{I}_s, \mathcal{I}_m)$. It is extremely difficult to find the maximum of the posterior in this form, so some manipulation is required. Using Bayes' rule [74],

$$\rho(A|B) = \frac{\rho(B|A)\rho(A)}{\rho(B)} \tag{2.8}$$

the posterior, $\rho(f_s|\mathcal{I}_s, \mathcal{I}_m)$, can be rewritten as the product of the likelihood, $\rho(\mathcal{I}_s, \mathcal{I}_m|f_s)$, and the prior probability, $\rho(f_s)$, to yield:

$$\rho(f_s|\mathcal{I}_s, \mathcal{I}_m) \propto \rho(\mathcal{I}_s, \mathcal{I}_m|f_s)\rho(f_s). \tag{2.9}$$

In the context of the stereopsis problem, given a candidate disparity map f_s, this equation can be evaluated as follows. The likelihood, $\rho(\mathcal{I}_s, \mathcal{I}_m|f_s)$, is evaluated by assuming that the observation noise in pixel intensities follows an independent identical distribution; which allows us to write:

$$\rho(\mathcal{I}_s, \mathcal{I}_m|f_s) = \prod_{\mathbf{p}\in\mathcal{I}_s} \rho(\mathcal{I}_s, \mathcal{I}_m|f_s(\mathbf{p})) \tag{2.10}$$

where $\rho(\mathcal{I}_s, \mathcal{I}_m|f_s(\mathbf{p}))$ is typically taken to be the exponential of the negative cost function that measures the similarity of pixels in $\mathcal{I}_s$ with pixels in $\mathcal{I}_m$ ($\rho(\mathcal{I}_s, \mathcal{I}_m|f_s(\mathbf{p})) = e^{-c(\mathbf{p}, f_s(\mathbf{p}))}$).

The prior, $\rho(f_s)$, is usually evaluated by assuming that the disparity function, f_s, is a Markov random field. So, the prior can be written as a product of joint probabilities:

$$\rho(f_s) = \prod_{\{\mathbf{p},\mathbf{q}\}\in\mathcal{N}_s} \rho_{\mathbf{p},\mathbf{q}}(f_s(\mathbf{p}), f_s(\mathbf{q})) \tag{2.11}$$

where $\mathcal{N}_s$ is the set of all four-connected neighbour pixel pairs in $\mathcal{I}_s$. $\rho_{\mathbf{p},\mathbf{q}}(f_s(\mathbf{p}), f_s(\mathbf{q}))$ gives the probability of having pixels $\mathbf{p}$ and $\mathbf{q}$ with disparities $f_s(\mathbf{p})$ and $f_s(\mathbf{q})$, respectively, given what is known about $\mathbf{p}$ and $\mathbf{q}$ – such as what their colours are, whether they are in the same colour segment, etc. Typically, $\rho_{\mathbf{p},\mathbf{q}}(f_s(\mathbf{p}), f_s(\mathbf{q}))$ takes the form of an exponential function, $\rho_{\mathbf{p},\mathbf{q}}(a, b) = e^{-V_{\mathbf{p},\mathbf{q}}(a,b)}$, for some function, $V_{\mathbf{p},\mathbf{q}}(a, b)$. Forms of the function $\rho_{\mathbf{p},\mathbf{q}}$ used by the research community are discussed later in Section 2.4.3.

Putting it all together, we arrive at the posterior probability for the stereo correspondence problem:

$$\rho(f_s|\mathcal{I}_s, \mathcal{I}_m) \propto \prod_{\mathbf{p}\in\mathcal{I}_s} \rho(\mathcal{I}_s, \mathcal{I}_m|f_s(\mathbf{p})) \prod_{\{\mathbf{p},\mathbf{q}\}\in\mathcal{N}_s} \rho_{\mathbf{p},\mathbf{q}}(f_s(\mathbf{p}), f_s(\mathbf{q})) \tag{2.12}$$

Energy Function Formulations

The energy minimization formulation for stereopsis is derived from the Bayesian probability formulation presented in the previous section. Rather than maximizing a probability we seek to minimize an energy function. Starting with Equation 2.9, and assuming that the energy follows a Gibbs distribution ($\rho(x) = e^{-E(x)}$ where $E(x)$ is the energy of the variable x) yields the energy formulation:

$$e^{-E(f_s|\mathcal{I}_s, \mathcal{I}_m)} \approx e^{-E(\mathcal{I}_s, \mathcal{I}_m|f_s)} e^{-E(f_s)} \tag{2.13}$$

which can be simplified by taking the negative logarithm of both sides; resulting in:

$$E(f_s|\mathcal{I}_s, \mathcal{I}_m) \approx E(\mathcal{I}_s, \mathcal{I}_m|f_s) + E(f_s). \tag{2.14}$$

In this equation, $E(\mathcal{I}_s, \mathcal{I}_m|f_s)$ is referred to as the *data term* and $E(f_s)$ is the *prior* or *smoothing* term. The data term gives the energy contribution of the matches imposed by the disparity map, f_s, on $\mathcal{I}_s$ and is often taken to be the sum of match cost errors over the pixels in $\mathcal{I}_s$:

$$E(\mathcal{I}_s, \mathcal{I}_m|f_s) = \sum_{\mathbf{p}\in\mathcal{I}_s} c(\mathbf{p}, f_s(\mathbf{p})) \tag{2.15}$$

The prior term gives the energy contribution from our knowledge of what the disparity map should look like. For instance, a very common form of this term is the Potts-energy smoothing term:

$$E(f_s) = \sum_{\{\mathbf{p},\mathbf{q}\} \in \mathcal{N}_s} V_{\mathbf{p},\mathbf{q}}(f_s(\mathbf{p}), f_s(\mathbf{q})) \tag{2.16}$$

where $\mathcal{N}_s$ is the set of all four-connected neighbour pixel pairs in $\mathcal{I}_s$ and

$$V_{\mathbf{p},\mathbf{q}}(a, b) = \lambda \min\{1, |a - b|\} \tag{2.17}$$

with λ being a user-supplied constant. Other options for the prior term are discussed later in Section 2.4.3.

Variant Formulations

In the previous two sections we presented the basic probabilistic and energy formulations of the stereo correspondence problem. Although most global stereo algorithms use one of these two formulations directly, some algorithms slightly alter the formulations. For instance, Kolmogorov and Zabih [57, 59, 60] and Goldlücke and Magnor [37] use energy minimization formulations that find disparity maps for more than one image simultaneously; Cheng and Caelli [20] propose a global stereo formulation that maximizes the joint posterior probability $\rho(f_s, \theta | \mathcal{I}_s, \mathcal{I}_m)$, where θ is the set of user-defined parameters in their probability formulation; and Vogiatzis et al. [94] formulate their solution as one of classifying three dimensional voxels as either inside or outside of the surface of the single object depicted in the input images.

Recently, there has also been a trend to formulate stereopsis algorithms over connected components of pixels rather than single pixels; these segment-based stereopsis algorithms assume that each connected component of pixels should have the same, or similar, disparity assignments and so they assign a disparity plane to each connected component rather than disparity values to each pixel individually.

Kolmogorov and Zabih [57, 59, 60] propose three slightly different global formulations of the stereo correspondence problem for assigning disparity values to multiple images at the same time. Their first formulation [57] defines an energy function over a set of *interactions* between pixels. They let $\mathcal{A}$ be the set of all pairs of pixels that can potentially correspond (i.e. $\{\mathbf{p}, \mathbf{q}\} \in \mathcal{A}$ if and only if there exists a δ such that $\mathbf{p}$ warps to $\mathbf{q}$ at disparity δ), and $f : \mathcal{A} \to \{0, 1\}$ be a configuration of $\mathcal{A}$ where $f(a) = 1$ indicates that interaction $a = \{\mathbf{p}, \mathbf{q}\}$ is *active* – that is, $\mathbf{p}$ and $\mathbf{q}$ warp to each other under their current disparity assignments. They also define an energy equation over f,

$$E(f) = E_{data}(f) + E_{occ}(f) + E_{smooth}(f)$$

where $E_{data}(f)$ is the sum of match costs for active interactions in $\mathcal{A}$, $E_{occ}(f)$ a penalty term of active interactions that are occluded by other active interactions, and $E_{smooth}(f)$ a smoothing term that penalizes neighbouring interactions ($\{\mathbf{p}, \mathbf{q}\}$ and $\{\mathbf{p}', \mathbf{q}'\}$ are neighbours if $\mathbf{p}$ and $\mathbf{p}'$ are four-connected neighbours in the same image) if the two interactions have the same disparity but only one of them is active. The stereo correspondence problem is then formulated as finding the function f that minimizes this energy equation with no pixel $\mathbf{p}$ in two active interactions.

For their second global stereo correspondence formulation, Kolmorogov and Zabih [59] use an energy minimization formulation that is similar to the standard form presented in the previous section. Given a set of input images, $\mathbb{I}$, they define the energy equation $E(\mathbb{F}|\mathbb{I})$ as:

$$E(\mathbb{F}|\mathbb{I}) = E_{data}(\mathbb{I}|\mathbb{F}) + E_{smoothness}(\mathbb{F}) + E_{visibility}(\mathbb{F})$$

where $\mathbb{F} = \{f_i : \mathcal{I}_i \in \mathbb{I}\}$ is the set of disparity maps defined on the images in $\mathbb{I}$. The data and smoothing terms are defined exactly as the data and prior terms in the standard energy minimization formulation presented in the previous section, except that they are defined over multiple disparity maps simultaneously. The visibility term, $E_{visibility}(\mathbb{F})$, sets the energy of a disparity assignment to infinity if any of the disparity assignments violate the weak consistency constraint [39].

Kolmogorov and Zabih's third global stereo formulation [60] is simply a generalization of their first two formulations. They combine the two energy equations into a single equation and add a consistency term that sets the energy to infinity if the set of active interactions and the disparity assignments in $\mathbb{F}$ are not consistent.

Cheng and Caelli [20] formulate the stereo correspondence problem as one of maximizing the posterior probability $\rho(f_s, \theta | \mathcal{I}_s, \mathcal{I}_m)$ where f_s is the disparity map of $\mathcal{I}_s$ and θ is the set of parameters used in their formulation. By formulating the stereo correspondence problem in this way they relieve the user from having to manually select a set of parameters that yield good results. Rather, they automatically learn a set of good parameters along side the disparity map.

Vogiatzis et al. [94] present a formulation of the stereo correspondence problem that finds the three dimensional surface of a single object depicted in the set of input images. They define the solution of the problem as the minimum of a binary energy function, the form of which is the same as that presented in the previous section, defined over three dimensional voxels; every voxel is either inside or outside of the surface. By defining the stereo correspondence problem as the minimum of a single binary energy function, Vogiatzis et al. are able to find the global minimum to their energy function via graph cuts minimization [59].

2.4.2 Methods of Solution

Given a global formulation of stereopsis there have been many different methods employed over the years to find an acceptable solution to the formulation. At the very least the formulations presented in Section 2.4.1 are NP-hard to find a global solution for, and in some cases finding a global solution is even NP-complete [57]. In this section we briefly discuss some of the methods that have been used by researchers to find approximate solutions to these formulations.

Early researchers, such as Lee et al. [62], use simulated annealing to find an approximate solution to their global formulations. Since simulated annealing is an energy minimization method, probability maximization formulations need be converted to an energy minimization formulation before an approximate solution can be found; this is usually done by assuming that the energy follows a Gibbs distribution, as in Section 2.4.1. However, simulated annealing can require tens, if not hundreds, of thousands of iterations to find an approximate solution to an energy equation. It comes as no surprise, then, that researchers have been experimenting with other approximation methods over the past eight, or so, years to improve the accuracy and running time of their algorithms.

Barbu and Zhu [3] propose using generalized Swendsen-Wang cuts to improve the convergence rate of simulated annealing. Generalized Swendsen-Wang cuts is a method for choosing which state to transition to in each iteration of simulated annealing based on the energy function being minimized. Rather than choosing a single pixel to change the disparity assignment of, as would be done with a Gibbs sampler, Swendsen-Wang cuts chooses a connected component, V_0, of pixels that have the same disparity value in the current state; V_0 is chosen probabilistically in a manner dictated by the energy function being minimized. Once V_0 has been chosen, a single new disparity value, that is generated by either a uniform random distribution or by a distribution dictated by the energy function, is assigned to all pixels in the component. By using this method, the simulated annealing algorithm is sped up by several orders of magnitude; in one experiment, the Swendsen-Wang cuts found a solution to an image segmentation problem approximately 500 times faster than a Gibbs sampler [3].

Gong and Yang [38] use a genetic algorithm to find an approximate solution to a standard energy formulation of stereopsis. For the genetic algorithm the image is represented as a quad-tree where all of the pixels in a leaf node of the tree are assigned the same disparity; if different disparity values are to be assigned to pixels in the same leaf, then the leaf is recursively split as required. All of the genetic algorithm operations (crossover, mutation, and splitting) are defined on this quad tree representation. In their experiments, the genetic algorithm requires almost a thousand generations to converge to an acceptable solution.

Many authors [15, 18, 33, 34, 39–41, 45, 63, 110] have experimented with using dynamic programming methods to find approximate solutions to their global stereopsis formulations. Dynamic programming methods have proven to be extremely fast, and amenable to near and real-time GPU-based implementations. Most dynamic programming methods for stereo correspondence find their solutions by minimizing the energy equation being used one scanline at a time; the approximate solution for each scanline is then merged into a single approximate solution over the whole image. However, this method tends to produce results that contain streaks in the direction of the dynamic programming optimization (along the scanlines).

In an effort to reduce the streaks that are a result of using dynamic programming optimization, Gong and Yang [41] and Lei et al. [63] both propose techniques for using dynamic programming that aim to reduce streaking. Gong and Yang propose an orthogonal reliability-based dynamic programming method. The disparity map for a given reference image is calculated in two stages. In the first stage, a dynamic programming optimization is performed over horizontal scanlines and those disparity assignments which are above a predefined reliability threshold (that is, the difference in match costs between the disparity assignment and the next best disparity assignment is above a threshold) are kept. In the second stage, the same dynamic programming optimization method is performed over vertical scanlines using the assigned disparities from the previous stage as ground control points (values that cannot be changed); again, only the disparity assignments with a reliability above a threshold are kept. The result is a non-dense disparity map (not every pixel is assigned a disparity value) that does not suffer from the streaking problem common in other dynamic programming based methods. Lei et al. [63] construct a special minimal spanning tree, the *region tree*, over colour segments in the reference image. The region tree is used to define an energy minimization formulation, and as the structure on which the dynamic programming is performed. The result is a streak free dense disparity map for the reference image.

The graph cuts technique for minimizing certain energy functions was first introduced to the stereo community by Boykov et al. [11]. Given a binary energy function, a special weighted graph where the vertices correspond with the variables in the energy function can be constructed such that the location of the min cut on the graph will tell us the variable assignment of the global minimum of the energy function. Boykov et al. leverage this feature of the graph cuts algorithm to find the global minimum of an energy function on binary variables to construct two iterative algorithms for finding a local minimum of energy functions over non-binary variables. Boykov et al. also prove that both of these algorithms will find a solution with energy that is within a constant multiple of the global minimum energy; however, the value of this particular result is questionable in light of the works of Tappen and Freeman [90] and Barbu and Zhu [3] that show, among other things, that the energy of the true disparity map is usually higher than that of the solution found by graph cuts.

The $\alpha\beta$ swap algorithm formulates the minimization process as a series of $\alpha\beta$ swaps, and if the prior term is at least a semimetric will guarantee that there is no lower energy disparity assignment within one $\alpha\beta$ swap of the reported solution [11] at convergence. An $\alpha\beta$ swap allows pixels that are assigned a disparity of α to either remain the same or be changed to β, and vice versa. At each iteration of the $\alpha\beta$ swap algorithm, a pair of disparities, α and β, are chosen (by random, in sequence, or by some other means) and the lowest energy disparity map within a single $\alpha\beta$ swap of the current disparity map is found via a binary graph cut. As presented, the algorithm continues iterating until a disparity map is found such that no single $\alpha\beta$ swap will yield a lower energy disparity map; in practice, though, the algorithm is halted after a fixed number of iterations.

The α expansion algorithm is similar to the $\alpha\beta$ swap algorithm. In this case, the minimization process is formulated as a series of α expansion moves wherein any pixel is allowed to take on the disparity value α, but no other changes are allowed. Similar to the $\alpha\beta$ swap algorithm, this algorithm guarantees that if the prior term is a metric then there is no other disparity map within a single α expansion move with lower energy than the solution found [11] at convergence. At each iteration of this algorithm a single disparity value, α, is selected and the lowest energy disparity map within a single α expansion of the current one is found by a single binary graph cut. Similar to the $\alpha\beta$ swap algorithm, the proposed α expansion algorithm continues iterating until there is no α expansion that will yield a lower energy disparity map. Again, in practice the algorithm is halted after a fixed number of iterations.

Kolmogorov and Zabih [59] characterize the energy functions over binary variables that can be minimized by graph cuts. A necessary condition for binary energy functions to be minimized by graph cuts, a sufficient condition for when energy functions that can be written as the sum of functions of up to three variables can be minimized by graph cuts, and a general method for constructing the graph used in the graph cuts method for a given energy function are presented. This work drastically increases the accessibility of the graph cuts algorithm to the stereo correspondence community (previously, the graphs used by the graph cuts algorithm had to be custom tailored to each energy function; which is not an easy task), and resulted in many authors [1,10,24,37,47,53,56–58,60,93,94,97] using graph cuts in their stereo correspondence algorithms.

Loopy belief propagation [74] has also been used by a number of authors [14,20,32,55, 64,89,104,106] to great effect. As of this writing, seven of the top ten reported algorithms in the Middlebury stereo rankings [80] use belief propagation to find an approximate solution to their stereo correspondence formulations; though, whether that is a function of the effectiveness of belief propagation or the energy/probability formulations used has not been tested.

Iterated conditional modes (ICM) optimization was applied to minimize stereo correspondence energy functions, defined over pixels and over pixel regions, by Jodoin and Mignotte [50]. The ICM algorithm is a greedy algorithm that iterates over pixels (or pixel regions when the energy function is defined over pixel regions) and chooses a disparity assignment for the pixel (pixel region) that gives the largest decrease in the value of the energy function; the iteration is continued until the energy of the current disparity map cannot be further decreased in this manner.

2.4.3 Prior Terms

In global formulations of the stereopsis problem the prior term is used to encourage the algorithm to produce disparity maps where neighbouring pixels with similar colour end up with the same, or similar, disparity values. Many different forms for the prior term have been used by various authors with no clear consensus on which form of the term is better under a given set of circumstances, and why. In this section we present several of the prior terms that have been used over the years by the research community.

Recall Equation 2.11 which states that the prior probability can be written as a product of joint probabilities:

$$\rho(f_s) = \prod_{\{\mathbf{p},\mathbf{q}\}\in\mathcal{N}_s} \rho_{\mathbf{p},\mathbf{q}}(f_s(\mathbf{p}), f_s(\mathbf{q}))$$

where $\mathcal{N}_s$ is the set of all four-connected neighbour pixel pairs in image $\mathcal{I}_s$. The difference between prior terms comes in the definition of the function $\rho_{\mathbf{p},\mathbf{q}}$.

The simplest forms for the function $\rho_{\mathbf{p},\mathbf{q}}$ are the linear and truncated linear models. Both of these models penalize neighbouring pixels that have different disparity values proportional to the difference of their disparity values. In the linear model, $\rho_{\mathbf{p},\mathbf{q}}$ is defined as:

$$\rho_{\mathbf{p},\mathbf{q}}(a,b) = e^{-\lambda|a-b|} \tag{2.18}$$

where λ is a user-defined constant. This model encourages the resulting disparity map to stray away from large disparity differences between neighbouring pixels; which may be a problem in the case of scenes containing foreground and background objects that have a large depth difference between them. As an alternative, some authors use a truncated linear model:

$$\rho_{\mathbf{p},\mathbf{q}}(a,b) = e^{-\lambda\min\{|a-b|,\tau_{\mathbf{pq}}\}} \tag{2.19}$$

where $\tau_{\mathbf{pq}}$ is a truncation value that can depend on the pixels $\mathbf{p}$ and $\mathbf{q}$. When $\tau_{\mathbf{pq}}$ is taken to be one for all $\mathbf{p}$ and $\mathbf{q}$, then this becomes the well known Potts model.

Kolmogorov and Zabih [57] introduce what has become a popular choice for the prior term. They define $\rho_{\mathbf{p},\mathbf{q}}$ in a manner based on the Potts model as follows:

$$\rho_{\mathbf{p},\mathbf{q}}(a,b) = e^{-\min\{1,|a-b|\}\times\begin{cases} c\lambda & \max\{(\vec{\mathbf{c}}(\mathbf{p})-\vec{\mathbf{c}}(\mathbf{q}))_i : i\} < \epsilon \\ \lambda & \text{otherwise} \end{cases}} \tag{2.20}$$

where c is a user-defined constant ($c = 3$ seems to be a popular choice among many authors). This model incorporates rudimentary edge detection to penalize neighbouring pixels with different disparities more heavily if the colours of the two pixels are very similar.

Barbu and Zhu [3] chose to incorporate the match cost function, $c()$, for $\mathbf{p}$ and $\mathbf{q}$ into their definition of $\rho_{\mathbf{p},\mathbf{q}}$:

$$\rho_{\mathbf{p},\mathbf{q}}(a,b) = 1 - e^{-\frac{\beta_{\mathbf{pq}}}{\alpha_1(c(\mathbf{p},a)+c(\mathbf{q},b))+\alpha_2}} \tag{2.21}$$

where $\beta_{\mathbf{pq}}$ is a constant that may depend on $\mathbf{p}$ and $\mathbf{q}$, and α_1 and α_2 are user-defined constants ($\alpha_1 = 3$ and $\alpha_2 = 10$ in their experiments).

Similarly, Jodoin and Mignotte [50] define $\rho_{\mathbf{p},\mathbf{q}}$ as:

$$\rho_{\mathbf{p},\mathbf{q}}(a,b) = p(a,b)e^{-\frac{||\vec{\mathbf{c}}(\mathbf{p})-\vec{\mathbf{c}}(\mathbf{q})||}{\beta^2}} \tag{2.22}$$

where $p(a,b)$ is a robust function such as the Leclerc function ($p(a,b) = 1 - e^{-(\frac{(a-b)}{\psi})^2}$ where ψ is a user-defined constant), and β is a user-defined constant.

Sun et al. [89] propose using a slightly different form of the prior term which is defined as the product of a smoothing prior and a segmentation prior. The smoothing prior is simply the product on the right hand side of Equation 2.11, and the segmentation prior is defined as:

$$\rho_{seg}(f_s) = \prod_{\{\mathbf{p},\mathbf{q}\}\in\mathcal{N}_s} e^{-\sigma(\mathbf{p},\mathbf{q})} \tag{2.23}$$

where $\sigma(\mathbf{p},\mathbf{q})$ is zero if $\mathbf{p}$ and $\mathbf{q}$ are in the same colour segment (as determined by running a colour segmentation algorithm on $\mathcal{I}_s$), and is the user-defined constant λ otherwise. Incorporating a segmentation prior such as ρ_{seg} allows their formulation to amplify the penalty imposed by the smoothing prior for pixels with different disparity labels when those pixels are in the same segment.

2.5 Summary

Stereopsis is the problem of inferring depth from binocular stereo sensory input. Given a binocular pair of stereo images, computational stereopsis algorithms aim to solve this problem through a number of different means. Although algorithms for solving the stereopsis problem come in many different flavours, there are commonalities among them: they all require a function (a match cost function) for measuring similarity between two image pixels and can be broadly categorized into either local or global algorithms. Local algorithms aim to solve the problem via a local analysis at each input-image pixel, whereas global algorithms formulate the stereopsis problem as one of finding an optimal solution to a global energy, or probability, function.

Chapter 3

Colour for Computational Stereopsis

Psychologists have been investigating the use of chromatic and achromatic mechanisms for stereopsis in the human visual system since at least the early 1960's [52]. Many experiments have been performed over the years with a general consensus forming that human stereopsis is performed using both chromatic (colour) and achromatic (luminance/brightness) mechanisms – though, it is still unclear exactly how these mechanisms are used in concert.

Jordan et al. [52] describe a system in which subjects are asked to determine the minimum disparity at which a wallpaper stereogram appears to flip in rank order from either ascending or descending from left to right. They call this minimum disparity the *rank-order disparity range*, and use it as an indicator of how efficient/capable a given sensory cue is at resolving depth; more efficient cues are assumed to be capable of resolving depth at larger disparities. Using this system they perform a number of experiments to determine the relative efficiency of chromatic and achromatic cues in the human visual system. Their experiments indicate that chromatic cues are more effective in extending the range at which depth can be inferred, chromatic and achromatic cues are used at about the same efficiency, chromatic cues are more effective against darker backgrounds, and that the human visual system is reluctant to fuse features with opposite contrasts.

Simmons and Kingdom [86, 87] hypothesize that stereopsis in humans results from a combination of the achromatic response, its gradient, and the chromatic response from the visual system. They also provide further evidence that achromatic or chromatic contrasts that are not correlated between the two eyes will severely disrupt stereopsis; for example, if an object appears bright in one eye and dark in the other.

den Ouden et al. [26] provide further evidence that chromatic information is used in stereopsis, but the data also suggests that chromatic and achromatic information is used independently in stereopsis – that is, the visual system seems to be able to determine whether the achromatic or chromatic mechanism is more accurate.

Thus, when using the human visual system as inspiration for stereopsis algorithms, it is only natural that researchers would use colour images in their algorithms. However, when using colour in a computer program there are many different representations of colour that can be used – each with their own benefits and drawbacks. In this Chapter we present the two prevailing models of human colour vision, and discuss many different mathematical representations of colour that are used in computer graphics.

3.1 Models of Colour Vision

The human retina contains four different types of nerve cells that are responsible for vision. One of these, the rods, are sensitive to only achromatic stimulus and are responsible for low-light vision. The other three, the cones, each respond to stimulus from different wavelengths of light and are responsible for colour vision; one responds to short (blue/violet) wavelength light, one to medium (green) wavelength light, and the last to long (red) wavelength light. Although the basic physiology of these nerve cells is fairly well understood, the precise mechanisms that transform the stimuli from these cells into the sensation of vision is still an active area of research. The two dominant models for the mechanism of colour vision are the Young-Helmholtz model and the Hering model.

In the Young-Helmholtz model [31,70,85,103], the stimuli from the cone photo-receptors in the eye are not transformed in any way. Thus, our brains directly combine the short, medium, and long wavelength stimuli to achieve the sensation of colour vision; just like mixing colours from the red, green, and blue primaries. For instance, white results from equal and simultaneous stimuli from all three cones, and yellow results from equal and simultaneous stimuli to the long (red) and medium (green) wavelength cones.

Part of what makes the Young-Helmholtz model popular is its simplicity – anyone who understands how to mix colours from the red, green, and blue primaries can understand the model. However, this simplicity prevents the model from explaining some visual phenomena. For instance, it cannot explain why we never see green reds or blue yellows. Nor does it explain why partial colour blindness only occurs as yellow-blue or red-green colour blindness.

The Hering model [49] of colour vision is based on the concept of an opponent-process. An opponent-process is basically a linear exclusive relationship between two colours; the two elements in an opponent-process are considered to be opposite and will not occur together. For example, in a yellow-blue opponent-process a positive response equates to a yellow response, with zero blue component, whereas a negative response equates to a blue response, with zero yellow component.

In the Hering model, vision is modeled using three opponent-processes: a yellow-blue, a red-green, and a white-black process. The model dictates that there is some neurological process between the eye and the brain that transforms the cone stimuli into these three opponent-process stimuli; colour is then the result of combining these three opponent-process stimuli. The yellow-blue opponent-process results from summing the long and medium wavelength cone stimulus responses and subtracting the short wavelength cone response, the red-green opponent-process is the result of subtracting the medium wavelength cone stimulus response from the long wavelength cone stimulus response, and the white-black opponent-process results from summing all three cone stimulus responses.

The Hering model is able to explain some of the visual phenomena that Young-Helmholtz is unable to. The red-green and yellow-blue opponent-responses explain why we never see green reds, red greens, yellow blues, or blue yellows – those colour pairs are opposite, and cannot occur together in this model. This model also explains red-green and yellow-blue colour blindness. Red-green colour blindness results from a deficiency in either the long or medium wavelength cones, and yellow-blue colour blindness results from a deficiency in the short wavelength cones.

Another difference between the two models is what colour they predict will be observed when there is no active stimulus to the photo-receptor nerves in the retina. In the Young-Helmholtz model, the mixture of three null responses from the cone photo-receptors results in the perception of the colour black. However, in the Hering model, this same condition results in a neutral grey sensation perception that is common after a long stay in darkness [49]. Furthermore, in the Hering model, "blackness arises neither by direct light stimulation nor in the simple absence of light, but rather by way of either simultaneous or successive contrast during, or following, light stimulation of some part of the retina." [49]

3.2 Mathematical Colour Representations

Although psychologists are concerned with accurately modeling how colour is represented and used in the human visual system, those of us in computer graphics related fields are free to use whatever mathematical representation of colour best serves our goals. In the case of computer vision fields, such as stereopsis, the differences in the relative positions, and distance between, colours in the space that is induced by each mathematical representation of colour can alter the resulting accuracy of algorithms.

For example, some representations represent colour in a perceptually uniform manner (CIELuv, CIELab); that is, the Euclidean distance between two colours in the colour space is an approximate measure of the relative perceptual difference between them. Other representations, such as those that result from the Karhunen-Loève transform [42], aim to transform the colour space such that each axis of the resulting space is orthogonal with respect to the information they encode about colour. Exactly which properties of a colour space are beneficial to different computer vision algorithms is not very well understood.

Note that throughout this dissertation we use the terms *colour representation* and *colour space* interchangeably; when using the phrase "colour space" we mean the vector space induced by a colour representation. For example, the RGB colour space is the vector space induced by the RGB colour representation.

In this section, we present a sampling of some of the colour representations in use by the computer graphics and computer vision communities.

3.2.1 RGB and sRGB

The RGB colour representation is the standard that most of the general public is familiar with; it is used to represent colour for computer screens, digital camera sensors, etc. In the RGB colour space, a colour is represented as a triple $\vec{\mathbf{c}} = (r, g, b)$ where $r, g, b \in [0, 1]$. The components of a colour in RGB space represent the amount of the respective primary colour (r for red, g for green, b for blue) that are combined to create the colour. Since colours expressed in RGB represent a mixture of the red, green, and blue primaries the RGB colour space is additive; that is, given two colours in RGB, $\vec{\mathbf{c^1}} = (r_1, g_1, b_1)$ and $\vec{\mathbf{c^2}} = (r_2, g_2, b_2)$, the sum $\vec{\mathbf{c^3}} = (\min\{r_1 + r_2, 1.0\}, \min\{g_1 + g_2, 1.0\}, \min\{b_1 + b_2, 1.0\})$ is the colour we would expect to obtain by mixing equal quantities of $\vec{\mathbf{c^1}}$ and $\vec{\mathbf{c^2}}$.

The sRGB [88] colour representation (officially the IEC 61966-2-1:1999 standard) is similar to the RGB representation in that it also represents colour using a triple $\vec{\mathbf{c}} = (r, g, b)$ with $r, g, b \in [0, 1]$. However, the sRGB standard was designed to explicitly compensate for the differences in the gamma-response of display (CRTs, LCDs, projectors, et cetera) and capture (digital cameras, et cetera) technology – these devices have exponential, rather than linear, voltage response curves that will cause images created with one device to look different on another unless the gamma-response is compensated for. The result of the sRGB standard is that regardless of what technology was used to create the image, it will look the same on every display device.

The device independence of the sRGB standard is obtained by encoding the components of the RGB colour using an exponential curve such that the components assume a device gamma (the exponent in the non-linear voltage response) of 2.2. That is, given a colour, $\vec{\mathbf{c}} = (r, g, b)$, in the RGB space the sRGB representation is calculated as $\vec{\mathbf{c'}} = (r', g', b')$ with:

$$c' = \begin{cases} 12.92 \times c & \leq 0.00304 \\ 1.055 \times c^{1.0/2.4} - 0.055 & \text{otherwise} \end{cases} \tag{3.1}$$

where $c \in \{r, g, b\}$. Since this encoding is non-linear, the colour space induced by the sRGB representation is not additive.

3.2.2 Standards by The International Commission on Illumination

The International Commission on Illumination (CIE), based out of Vienna, Austria, is dedicated to the advancement of colourimetric research, and provides many standards related to colourimetry. In this section we discuss some of the colour representation standards they have created.

CIEXYZ

The CIE 1931 2° standard observer, also commonly known as CIEXYZ, was standardized in 1931 from measurements of human vision taken through a 2° viewing angle. It serves as a standardized RGB colour space where the X, Y, and Z components of a colour, $\vec{\mathbf{c}} = (X, Y, Z)$ with $X, Y, Z \in [0, 1]$, in the CIEXYZ space very loosely correspond with the red, green, and blue components of the colour, respectively. Additionally, the Y component of the colour is designed to provide a measure of the luminance of the colour. This colour representation serves as the basis for many of the other standard colour representations by the CIE.

Given a colour, $\vec{\mathbf{c}}$, represented by its spectral power distribution (power per unit area per unit wavelength), $I(\lambda)$, the X, Y, and Z components of the CIEXYZ representation of $\vec{\mathbf{c}}$ are defined as:

$$X = \int_0^\infty I(\lambda)X(\lambda)d\lambda \tag{3.2a}$$

$$Y = \int_0^\infty I(\lambda)Y(\lambda)d\lambda \tag{3.2b}$$

$$Z = \int_0^\infty I(\lambda)Z(\lambda)d\lambda \tag{3.2c}$$

where $X(\lambda)$, $Y(\lambda)$, and $Z(\lambda)$ are the CIE 1931 2° standard observer colour matching functions depicted in Figure 3.1.

However, we rarely have the spectral power distribution of a colour. So, the CIE has also defined a transform for converting the RGB representation of a colour to its CIEXYZ representation. Given the RGB representation of a colour, $\vec{\mathbf{c}} = (r, g, b)$, that is assumed to have been captured under the conditions of CIE D65 white point illumination (roughly equivalent to mid-day light), the CIEXYZ representation of $\vec{\mathbf{c}}$ is defined as:

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} 0.412453 & 0.357580 & 0.180423 \\ 0.212671 & 0.715160 & 0.072169 \\ 0.019334 & 0.119193 & 0.950227 \end{pmatrix} \begin{pmatrix} r \\ g \\ b \end{pmatrix} \tag{3.3}$$

CIExyY

Colour can be divided into two parts: luminance, and chromaticity. The luminance is a measure of the intensity of the colour, and the chromaticity is an indication of the hue of the colour. The CIExyY colour representation is derived from the CIEXYZ representation to explicitly divide colours into these two constituent parts. Given the CIEXYZ representation of a colour, $\vec{\mathbf{c}} = (X, Y, Z)$, the CIExyY representation of the colour, (x, y, Y), is calculated as:

$$x = \frac{X}{X + Y + Z} \tag{3.4a}$$

$$y = \frac{Y}{X + Y + Z} \tag{3.4b}$$

$$Y = Y \tag{3.4c}$$

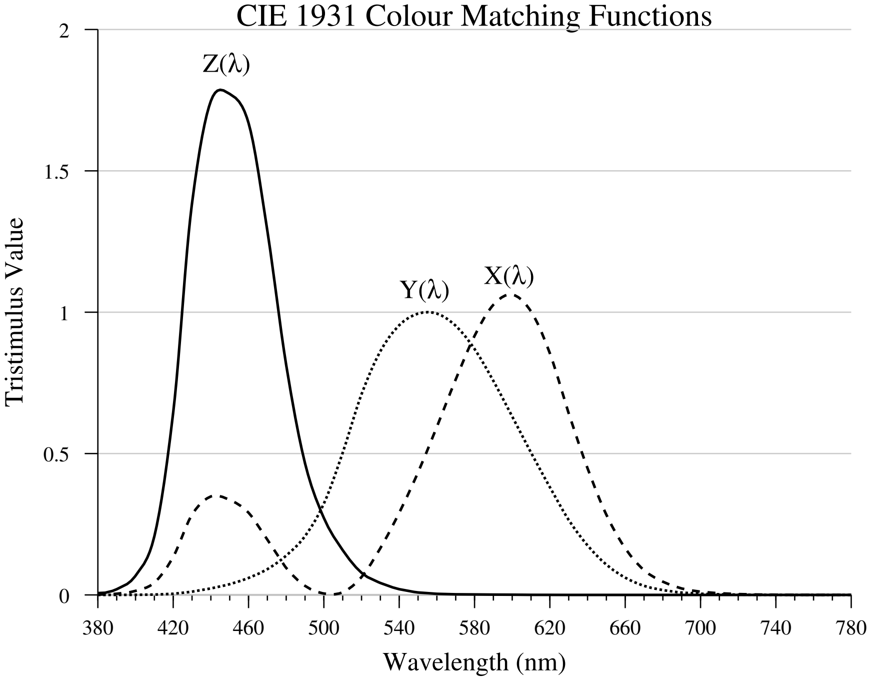


Figure 3.1: CIE 1931 colour matching functions.

In this representation, the x and y components of the colour specify the chromaticity, and the Y component is a measure of luminance.

CIELab

The CIELab colour representation (also known as CIEL*a*b*, or the CIE 1976 (L*a*b*) colour space) was standardized by the CIE in 1976 as a colour space that is more perceptually uniform than the, very nonuniform, CIEXYZ colour space. Like the CIExyY colour representation, the CIELab colour representation is a derivative of CIEXYZ that divides colours into their chromaticity and luminance. The L channel of a CIELab colour provides a measure of the luminance of the colour, while the a and b components provide an opponent-process representation of the chromaticity of the colour. The a component represents a red-green opponent-process, and the b component represents a yellow-blue opponent-process. Thus, the CIELab colour representation is a direct application of the Hering colour vision model to colour representations.

Given a colour represented in CIEXYZ as $\vec{\mathbf{c}} = (X, Y, Z)$, the CIELab representation of the colour, (L, a, b), is calculated as:

$$L = \begin{cases} 116 \times \sqrt[3]{Y} - 16 & \text{if } Y > 0.008856 \\ 903.3 \times Y & \text{otherwise} \end{cases} \tag{3.5a}$$

$$a = 500 \times (f(X/0.950456) - f(Y)) \tag{3.5b}$$

$$b = 200 \times (f(Y) - f(Z/1.088754)) \tag{3.5c}$$

where

$$f(t) = \begin{cases} \sqrt[3]{t} & \text{if } t > 0.008856 \\ 7.787 * t + \frac{16}{116} & \text{otherwise} \end{cases} \tag{3.6}$$

CIELuv

The CIELuv colour representation (also known as the CIE 1976 (L*u*v*) colour space) was also standardized in 1976 by the CIE as an alternative to CIELab for a perceptually uniform colour space. Both CIELab and CIELuv have their proponents and detractors, so rather than choosing just one standard colour space to release in 1976 the CIE released both. As with CIELab, CIELuv is a derivative of CIEXYZ that divides colours into their chromaticity and luminance.

The L component in a CIELuv colour representation is identical to the L channel in the CIELab representation; it provides a measure of the luminance of the colour. The u and v components represent the chromaticity of the colour, but, unlike the CIELab representation, they are not opponent-response components.

Given a colour, $\vec{\mathbf{c}} = (X, Y, Z)$, in the CIEXYZ colour space, the CIELuv representation of that colour, (L, u, v), can be calculated as:

$$L = \begin{cases} 116 \times \sqrt[3]{Y} - 16 & \text{if } Y > 0.008856 \\ 903.3 \times Y & \text{otherwise} \end{cases} \tag{3.7a}$$

$$u = 13L \times \left(\frac{4X}{X + 15Y + 3Z} - 0.19783943 \right) \tag{3.7b}$$

$$v = 13L \times \left(\frac{9Y}{X + 15Y + 3Z} - 0.46831096 \right) \tag{3.7c}$$

Furthermore, the subplanes of the CIELuv colour space defined by constant luminance (equal L values) are additive. That is, given two colours with the same luminance, the u and v components of those colours can be added to obtain the colour at the same luminance that one would expect to obtain by mixing the two colours.

3.2.3 YCrCb

YCrCb is a colour representation that is used in some video and digital image representations; it is the colour representation used in the jpeg file format [43], for instance. The Y component of a colour represented in YCrCb space is a measure of the luminance of the colour. The chromaticity of the colour is reflected in its Cr and Cb components. To reduce storage/bandwidth requirements of images using YCrCb, the Y components are often stored/transmitted at full/high bandwidth, while the Cr and Cb components are bandwidth-reduced/subsampled/compressed/et cetera. The nature of this colour representation is such that storing/transmitting YCrCb images in this manner does not have much effect on the visual quality of the image.

Given the RGB representation of a colour, $\vec{\mathbf{c}} = (r, g, b)$, the YCrCb representation of the colour, (Y, Cr, Cb), is calculated as:

$$Y = 0.299r + 0.587g + 0.114b \tag{3.8a}$$

$$Cr = 0.713(r - Y) + 0.5 \tag{3.8b}$$

$$Cb = 0.564(b - Y) + 0.5 \tag{3.8c}$$

3.2.4 HSV and HSVXYZ

The hue, saturation, value (HSV) colour space is commonly used by digital artists to simplify the process of selecting colours. The H component of an HSV colour is an angular representation of the hue of the colour, the S component indicates how saturated the colour is, and the V component encodes the luminance. Given the RGB representation of a colour, $\vec{\mathbf{c}} = (r, g, b)$, the HSV representation of the colour, (H, S, V), is calculated as follows:

$$V = \max\{r, g, b\} \tag{3.9a}$$

$$S = \begin{cases} \frac{V - \min\{r,g,b\}}{V} & \text{if } V \neq 0 \\ 0 & \text{otherwise} \end{cases} \tag{3.9b}$$

$$H = \begin{cases} 0 & S = 0 \\ \frac{60(g-b)}{S} & V = r, S \neq 0 \\ 120 + \frac{60(b-r)}{S} & V = g, S \neq 0 \\ 240 + \frac{60(r-g)}{S} & \text{otherwise} \end{cases}. \tag{3.9c}$$

Then, if $H < 0$ it is placed into the range $[0, 360)$ by adding 360.

Special care must be taken when manipulating and comparing colours in HSV space because the H component is an angular measure. Simply taking the difference of two HSV colours can result in the wrong values for the H component of the difference vector. For example, consider the difference between the colours $(1, 1, 1)$ and $(359, 1, 1)$ in HSV space; just taking their difference results in $(-358, 0, 0)$ whereas the two colours only differ by 2 degrees of hue, not 358. Thus, we calculate the difference between two colours, (h_1, s_1, v_1) and (h_2, s_2, v_2), in HSV space as:

$$\Delta S = s_1 - s_2 \tag{3.10a}$$

$$\Delta V = v_1 - v_2 \tag{3.10b}$$

$$\Delta H = \begin{cases} 0 & s_1 = 0, s_2 = 0 \\ s_1 - s_2 + 360 & s_1 - s_2 < -180 \\ s_1 - s_2 - 360 & s_1 - s_2 > 180 \\ s_1 - s_2 & \text{otherwise} \end{cases} \tag{3.10c}$$

Furthermore, the average of colours represented in HSV space is not well defined. Consider the average of $(60, 1, 1)$, and $(300, 1, 1)$; just adding and dividing by 2 results in $(180, 1, 1)$, though $(0, 1, 1)$ is the correct answer (since a hue angle of 0 is also halfway between 60 and 300). When an algorithm calls for the mean of a set of colours, we calculate the mean incrementally rather than summing all of the colour values and dividing by the number of colours in the set; the result is proper calculation of the mean around the 0 hue angle.

The HSV colour space can be visualized in three dimensions as a unit-height and unit-radius cone with its tip at the origin and axis aligned with the z-axis. The HSVXYZ space is the result of transforming colours from the HSV space to this cone representation. Given the HSV representation of a colour, $\vec{\mathbf{c}} = (H, S, V)$, we calculate the HSVXYZ representation of the colour, (x, y, z), as:

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} VS\cos\left(\frac{H\pi}{180}\right) \\ VS\sin\left(\frac{H\pi}{180}\right) \\ V \end{pmatrix} \tag{3.11}$$

3.2.5 HLS and HLSXYZ

The hue, luminance, saturation (HLS) colour space is similar to the HSV colour space, and is also used by digital artists for colour selection. The L component of a HLS colour is roughly equivalent to the V component of a HSV colour, while the S component is calculated differently. Given the RGB representation of a colour, $\vec{\mathbf{c}} = (r, g, b)$, the HLS representation of the colour, (H, L, S), is calculated as:

$$V_{max} = \max\{r, g, b\} \tag{3.12a}$$

$$V_{min} = \min\{r, g, b\} \tag{3.12b}$$

$$L = \frac{V_{max} + V_{min}}{2} \tag{3.12c}$$

$$S = \begin{cases} 0 & L = 0 \\ \frac{V_{max} - V_{min}}{V_{max} + V_{min}} & 0 < L < 0.5 \\ \frac{V_{max} - V_{min}}{2 - (V_{max} + V_{min})} & \text{otherwise} \end{cases} \tag{3.12d}$$

$$H = \begin{cases} 0 & S = 0 \\ \frac{60(g-b)}{S} & V = r, S \neq 0 \\ 120 + \frac{60(b-r)}{S} & V = g, S \neq 0 \\ 240 + \frac{60(r-g)}{S} & \text{otherwise} \end{cases} . \tag{3.12e}$$

Note that the same care must be taken when calculating colour differences, and averages in HLS space as is taken in HSV space.

As with the HSV colour space, the HLS colour space can be visualized in three-dimensions. Rather than a unit-radius cone, however, the HLS space can be visualized as a unit sphere centered at the origin. Given the HLS representation of a colour, $\vec{\mathbf{c}} = (H, L, S)$, the HSVXYZ representation of the colour is calculated as:

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} K \cos\left(\frac{H\pi}{180}\right) \\ K \sin\left(\frac{H\pi}{180}\right) \\ 2(L - 0.5) \end{pmatrix} \tag{3.13}$$

where $K = S \cos(2\pi(L - 0.5))$

3.2.6 $c_1c_2c_3$ and $l_1l_2l_3$

The $c_1c_2c_3$ and $l_1l_2l_3$ colour spaces were developed by Gevers and Stokman [35] for the purposes of edge detection. In their work, they show that the $c_1c_2c_3$ colour space is colour invariant for matte objects – it varies with change in material and highlights – and the $l_1l_2l_3$ colour space is invariant for matte and specular objects – it only varies with changes in material.

Given the RGB representation of a colour, $\vec{\mathbf{c}} = (r, g, b)$, the $c_1c_2c_3$ representation of the colour, (c_1, c_2, c_3), is calculated as:

$$c_1 = \begin{cases} \frac{\pi}{2} & b = 0, g = 0 \\ \arctan\left(\frac{r}{\max\{g,b\}}\right) & \text{otherwise} \end{cases} \tag{3.14a}$$

$$c_2 = \begin{cases} \frac{\pi}{2} & b = 0, r = 0 \\ \arctan\left(\frac{g}{\max\{r,b\}}\right) & \text{otherwise} \end{cases} \tag{3.14b}$$

$$c_3 = \begin{cases} \frac{\pi}{2} & r = 0, g = 0 \\ \arctan\left(\frac{b}{\max\{r,g\}}\right) & \text{otherwise} \end{cases} \tag{3.14c}$$

and the $l_1 l_2 l_3$ representation of the colour, (l_1, l_2, l_3), is calculated as:

$$\begin{pmatrix} l_1 \\ l_2 \\ l_3 \end{pmatrix} = \begin{cases} \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} & \text{if } D = 0 \\ \begin{pmatrix} \frac{|r-g|}{D} \\ \frac{|r-b|}{D} \\ \frac{|g-b|}{D} \end{pmatrix} & \text{otherwise} \end{cases} \tag{3.15}$$

where $D = |r - g| + |r - b| + |g - b|$

3.2.7 $h_1 h_2 h_3$

The $h_1 h_2 h_3$ colour space is an opponent-colour space that was developed by Wesolkowski et al. [98] for use in edge detection. Colours are represented in this colour space using red-green (h_1), green-blue (h_2), and blue-red (h_3) opponent-responses. Given the RGB representation of a colour, $\vec{\mathbf{c}} = (r, g, b)$, the $h_1 h_2 h_3$ representation of the colour, (h_1, h_2, h_3), is:

$$\begin{pmatrix} h_1 \\ h_2 \\ h_3 \end{pmatrix} = \begin{pmatrix} r - g \\ g - b \\ b - r \end{pmatrix} \tag{3.16}$$

3.2.8 The Brockelbank Colour Space

The Brockelbank colour space is an opponent-colour space that conforms to the Hering model of colour vision. This representation of the Hering model was presented by Brockelbank and Yang [13] in their work on computational stereopsis algorithms. Colour is represented in this colour space by dark-light (r_1), red-green (r_2), and blue-yellow (r_3) opponent-process components. Given the RGB representation of a colour, $\vec{\mathbf{c}} = (r, g, b)$, the Brockelbank representation of the colour, (r_1, r_2, r_3), is:

$$\begin{pmatrix} r_1 \\ r_2 \\ r_3 \end{pmatrix} = \begin{pmatrix} r + g \\ r - g \\ r + g - b \end{pmatrix} \tag{3.17}$$

3.2.9 The Campbell Colour Space

The Campbell colour space was proposed in Campbell et al. [16] for image segmentation. It is another opponent-color space that is based on the Hering model of colour vision; the c_{lum} channel provides a measure of the brightness of the colour, c_{rg} is the red-green opponent-process component, and the c_{yb} component is the yellow-blue opponent-process component. Given the RGB representation of a colour, $\vec{\mathbf{c}} = (r, g, b)$, the Campbell colour space representation of the colour, $(c_{lum}, c_{rg}, c_{yb})$, is:

$$c_{lum} = \frac{3r + 6g + b}{10} \tag{3.18a}$$

$$c_{rg} = \frac{r - g + 1}{2} \tag{3.18b}$$

$$c_{yb} = \frac{r + g - 2b + 2}{4} \tag{3.18c}$$

3.2.10 The Karhunen-Loève Transform

The Karhunen-Loève transform [42] is a linear transform of a colour space that transforms the colour space such that it is optimal, in an information theoretic sense, for a particular image. That is, the components of the resulting colour space are orthogonal, and thus, do not contain any redundant information. Given an image, the Karhunen-Loève transform for that image is simply the transpose of the matrix of unit-length eigenvectors of the covariance matrix of the image; that is, the eigenvectors of the covariance matrix form rows in the transformation matrix.

Given an image, $\mathcal{I}$, with colour pixels that are expressed in some n-dimensional colour representation. The Karhunen-Loève transform corresponding to $\mathcal{I}$ is calculated as follows. First, the covariance matrix, $\mathbf{C}$, of the image pixel colours is calculated as:

$$\mathbf{C} = \frac{1}{||\mathcal{I}||} \sum_{\mathbf{p} \in \mathcal{I}} < \vec{\mathbf{c}}(\mathbf{p}) - \bar{\vec{\mathbf{c}}}, \vec{\mathbf{c}}(\mathbf{p}) - \bar{\vec{\mathbf{c}}} > \tag{3.19}$$

where $||\mathcal{I}||$ denotes the number of pixels in the image $\mathcal{I}$, $\bar{\vec{\mathbf{c}}} = \frac{1}{||\mathcal{I}||} \sum_{\mathbf{p} \in \mathcal{I}} c(\mathbf{p})$ is the mean colour over all pixels in the image, and $< \vec{\mathbf{a}}, \vec{\mathbf{b}} >$ denotes the $n \times n$ matrix formed by the outer product of the two n-dimensional vectors, $\vec{\mathbf{a}}$ and $\vec{\mathbf{b}}$:

$$< \vec{\mathbf{a}}, \vec{\mathbf{b}} >= \begin{pmatrix} \vec{\mathbf{a}}_1\vec{\mathbf{b}}_1 & \vec{\mathbf{a}}_1\vec{\mathbf{b}}_2 & \cdots & \vec{\mathbf{a}}_1\vec{\mathbf{b}}_n \\ \vec{\mathbf{a}}_2\vec{\mathbf{b}}_1 & \vec{\mathbf{a}}_2\vec{\mathbf{b}}_2 & \cdots & \vec{\mathbf{a}}_2\vec{\mathbf{b}}_n \\ \vdots & \vdots & \ddots & \vdots \\ \vec{\mathbf{a}}_n\vec{\mathbf{b}}_1 & \vec{\mathbf{a}}_n\vec{\mathbf{b}}_2 & \cdots & \vec{\mathbf{a}}_n\vec{\mathbf{b}}_n \end{pmatrix} \tag{3.20}$$

where $\vec{\mathbf{a}}_i$ denotes the i^{th} element of the vector $\vec{\mathbf{a}}$.

Since the matrix $\mathbf{C}$ is a real symmetric matrix it has n orthonormal eigenvectors. Given the n eigenvectors of $\mathbf{C}$, $\vec{\mathbf{E^1}}$, $\vec{\mathbf{E^2}}$, $\ldots$, $\vec{\mathbf{E^n}}$, the Karhunen-Loève transform is given by the matrix:

$$\mathbf{K} = \begin{pmatrix} \vec{\mathbf{E^1}}_1 & \vec{\mathbf{E^1}}_2 & \cdots & \vec{\mathbf{E^1}}_n \\ \vec{\mathbf{E^2}}_1 & \vec{\mathbf{E^2}}_2 & \cdots & \vec{\mathbf{E^2}}_n \\ \vdots & \vdots & \ddots & \vdots \\ \vec{\mathbf{E^n}}_1 & \vec{\mathbf{E^n}}_2 & \cdots & \vec{\mathbf{E^n}}_n \end{pmatrix}. \tag{3.21}$$

Given a Karhunen-Loève transformation matrix, $\mathbf{K}$, and a colour vector $\vec{\mathbf{c}}$, the Karhunen-Loève transform of $\vec{\mathbf{c}}$ is given by $\mathbf{K}\vec{\mathbf{c}}$.

3.2.11 The Ohta Colour Space

Ohta et al. [71] present a segmentation algorithm that recursively segments an input image using a histogram of colour features at each recursive step. In order to increase the discriminative power of the histogram for segmenting the image, they perform a Karhunen-Loève transform on the region of interest at each recursive step of the algorithm. However, calculating a full transform at each recursive step of their algorithm is costly, so they analyze the transforms computed when their algorithm is run on a set of six images to derive a single colour space that they can apply a single time to the input image rather than dynamically recalculating a colour transform at each recursive step. The Ohta colour space is the space they derive via this process.

Given the RGB representation of a colour, $\vec{\mathbf{c}} = (r, g, b)$, the Ohta space representation of the colour, (i_1, i_2, i_3), is:

$$\begin{pmatrix} i_1 \\ i_2 \\ i_3 \end{pmatrix} = \begin{pmatrix} \frac{r+g+b}{3} \\ r-b \\ \frac{2g-r-b}{2} \end{pmatrix} \tag{3.22}$$

3.3 Colour Representations Used in Computational Stereopsis Research

Although using colour spaces other than RGB is relatively common in computer vision algorithms for segmentation [21, 23, 29, 71], edge detection [35, 98] , texture classification [73], optical flow [111], and face detection [25] it is strangely uncommon in computational stereopsis algorithms.

Jordon and Bovik [51] use a normalized RGB space in their edge-matching stereopsis algorithm, where each colour is divided by the sum of the blue, green, and red intensities of the colour.

Brockelbank and Yang [13] compare the performance of an edge-matching stereopsis algorithm in a greyscale, RGB, and an opponent-colour space; concluding that, for their particular algorithm, optimal performance seems to be achieved by utilizing an opponent-colour space.

Koschan [61] performs a small-scale study on the performance of a winner-take-all (WTA) stereopsis algorithm utilizing five different match cost functions in the greyscale, RGB, CIEXYZ, HSV, and Ohta colour spaces; even though none of the data from the experiments is presented, Koschan states that the best results are always obtained when the Ohta colour space is used.

Bay et al. [4] use a quantization of the HSV cone colour space (HSVXYZ) to construct colour histograms on the left and right sides of line segments in the line-matching stage of their wide-baseline stereo matching algorithm.

Pinhasov et al. [76] present WTA stereopsis algorithms that use the sum-of-squared-differences (SSD) metric on locally and globally optimized colour space transforms. Their locally optimizing algorithm calculates an optimal transform of the RGB colour space for small patches of constant disparity, and uses the transformed space instead of the RGB space to perform the WTA algorithm. Their globally optimizing algorithm is similar to their locally optimizing algorithm with the exception that it calculates a single colour space transform for the entire image rather than an individual transform for each patch.

Hua et al. [48] present a stereopsis algorithm based on neural networks and evaluate its performance in seven different colour spaces: RGB, CIELab, CIEXYZ, YCbCr, Ohta, HLS, and an opponent-colour space. Comparing the results of running their algorithm on two stereo image pairs in these seven colour spaces, they find that their algorithm performs best when using the Ohta, YCrCb, and opponent-colour spaces.

Cai et al. [15] develop a stereopsis algorithm that uses fuzzy set theory in a matching process that is applied to scanlines of the input images. Their algorithm is developed to use input in the HSV colour space.

Yoon and Kweon [105] present the adaptive support weight match cost function for use in stereopsis algorithms. This match cost function aggregates disparity costs over a small window using the colour difference in CIELab space of the underlying pixels to weight contributions.

Zickler et al. [111] propose a novel rotation of the RGB cube that is performed independently on every pixel of an input image to derive a greyscale image that does not contain any of the specular highlights present in the input. They apply this transform to tasks such as stereopsis, optical flow, shape from shading, and photometric stereo. However, a major limiting factor of the method is that the lighting of the scene must be known to derive the rotation of the RGB cube.

3.4 Summary

Both chromatic and achromatic information are useful for computational stereopsis algorithms. However, there are many options for mathematically representing this information with no clear consensus on which is best, or even what properties of a representation are beneficial. There are two primary models of human vision that have been used as a basis for many of these colour representations – the Young-Helmholz trichromatic model, and the Hering model. Both of these models suggest that human vision uses input derived from a decomposition of colour into red, green, and blue primaries, but the Young-Helmholtz model suggests that this decomposition is used directly in human vision whereas the Hering model suggests that the information is first processed into a set of three (one achromatic and two chromatic) opponent-responses.

In computational stereopsis research, the standard RGB representation of colour is, by far, the most commonly used colour representation. Exactly why this is the case is not clear given the successes in utilizing other colour representations in other fields of computer vision, and the evidence provided by some computational stereopsis researchers that other colour representations can improve the accuracy of their algorithms.

Chapter 4

Evaluation of Stereopsis Algorithms

When evaluating stereopsis algorithms it is important that the algorithm is compared, either qualitatively or quantitatively, with algorithms that are as similar as possible to the algorithm to be evaluated. This sort of evaluation can enable researchers to form hypotheses as to why a given stereopsis algorithm performs better than another, rather than simply allowing them to conclude that algorithm X is better than Y with no indication as to why that is the case. For instance, if an author were proposing a new form for the prior function $\rho_{\mathbf{p},\mathbf{q}}$ in a global formulation then comparison should be made against algorithms that only differ in the definition of $\rho_{\mathbf{p},\mathbf{q}}$. Furthermore, the comparison should be performed on multiple image sets with diverse scene characteristics so as to increase the probability that the evaluation is indicative of the general performance of the algorithms being compared, rather than their performance over a small specialized input set.

Unfortunately, the structured evaluation of stereopsis algorithms that would allow for more insight into the causes of their relative performance is very rarely used among stereopsis researchers. In fact, prior to the introduction of a standardized data set and evaluation methodology by Scharstein and Szeliski [82] in 2002 evaluations of stereopsis algorithms tended to be very *ad hoc* – most using different data sets and/or evaluation methodology. The result of having different data sets and evaluation methodology used by stereopsis researchers is that it is generally impossible to compare the relative performance of stereopsis algorithms without implementing each algorithm and evaluating them yourself.

Furthermore, the overwhelming majority of evaluations performed in the research literature on stereopsis algorithms is not indicative of any real expectation users can have in their relative performance. These evaluations use a very small data set – typically less than five image pairs are used – to compare algorithms and methods, and do not result in an evaluation with any sort of statistical significance.

In this Chapter we present some of the more popular existing methods for evaluating stereopsis algorithms as well as our own methodology for comparing stereopsis algorithms.

4.1 Existing Methods of Evaluation

Evaluation of stereopsis algorithms has taken many forms over the years – both qualitative and quantitative. The quantitative evaluations provide an objective view of the relative performance of stereopsis algorithms, or their components, while qualitative evaluations allow a reader to visually identify potential problem areas of the given algorithm. In the area of stereopsis algorithm research, both approaches to evaluation can be quite useful. For instance, qualitative evaluations are the only available option when designing a stereopsis algorithm for an application where it is difficult, if not impossible, to obtain ground truth depth maps; such as in stereoscopic reconstruction from remote sensing data.

4.1.1 Data Sets Used

There are many options available for image data sets for stereopsis evaluation. Synthetic images offer several advantages over photographs (they are easy to create, have readily available ground truth, can be customized to test specific cases, et cetera), but care must be taken when using them since they may not represent the real photographs that would be encountered in practice. Options for synthetically created image sets include: two dimensional solid shapes over a solid background, stereograms, renderings of a three dimensional textured object, and renderings of complicated three dimensional scenes.

Photographic data sets, on the other hand, can be more difficult to obtain (especially when corresponding ground truth disparity maps are also required), but can more easily represent the reality in which a given stereopsis algorithm is to be used in practice. Fortunately, many photographic image pairs, with corresponding ground truth, have been made available by several researchers. Satoh and Ohta [78] provide colour images and ground truth for the Tsukuba head and lamp data set (often simply called the Tsukuba data set). Scharstein and his colleagues [46, 79, 82, 83] have made 38 image pairs (known as the Middlebury 2001 through 2006 data sets), with ground truth, available; the Middlebury 2002 data set along with the Venus pair from the Middlebury 2001 data set, and the Tsukuba data set are the four image pairs currently being used in the de facto standard evaluation method used by the stereopsis research community – the Middlebury stereo evaluation framework [80].

4.1.2 Quantitative Evaluation

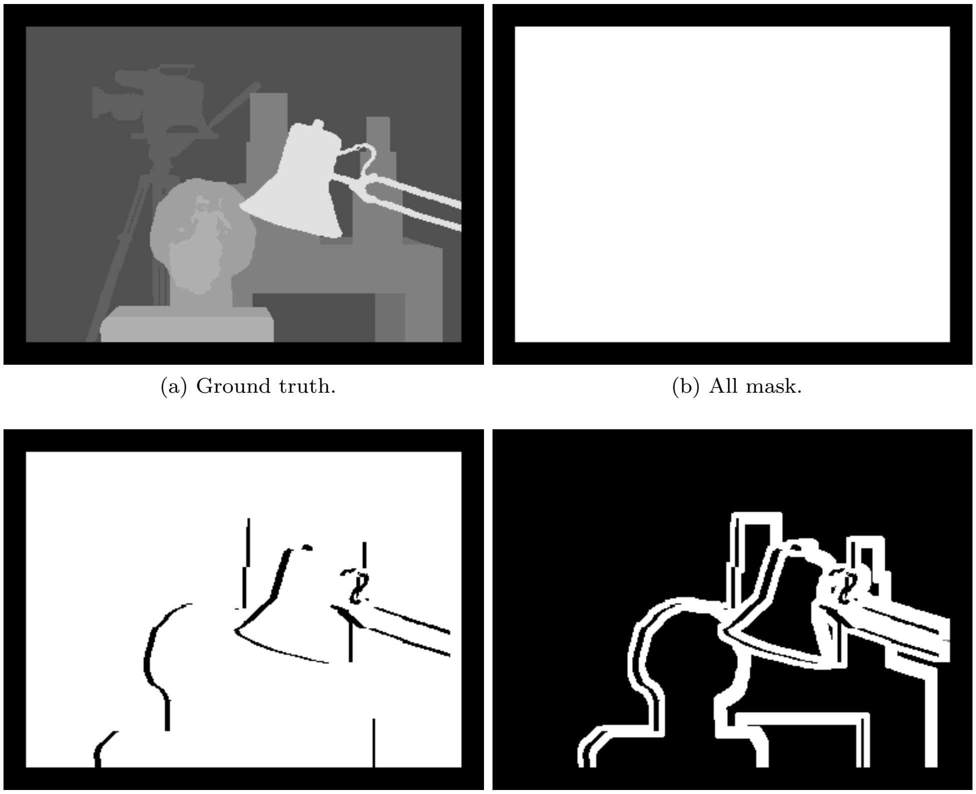
Many different forms of quantitative evaluation of stereopsis algorithms have been utilized over the years. In most cases, these quantitative evaluations require the ground truth depth/disparity map for the images being used in the evaluation. Though, some evaluations that are not concerned with the accuracy of the algorithms being compared do not have this requirement.

The most common quantitative evaluation metric used by stereopsis researchers is the percentage of pixels in the calculated disparity map that differ from the ground truth by more than a user-defined threshold. The popularity of this particular metric of performance can be attributed to Scharstein and Szeliski who chose to use it in their now de facto standard stereopsis evaluation framework [80, 82]. Given a disparity map, $\mathcal{I}_c$, to be evaluated and the ground truth disparity map, $\mathcal{I}_t$, the percentage of incorrectly assigned pixels in $\mathcal{I}_c$ is calculated as:

$$P = \frac{1}{N} \sum_{\mathbf{p}} T(|\delta_c(\mathbf{p}) - \delta_t(\mathbf{p})| \geq \tau) \tag{4.1}$$

where $T(\cdot)$ is the truth function ($T(true) = 1$, $T(false) = 0$), τ is the user-defined error threshold, $\delta_k(\mathbf{p})$ is the disparity of pixel $\mathbf{p}$ in $\mathcal{I}_k$, and N is the number of pixels in $\mathcal{I}_c$.

However, Equation 4.1 is rarely used in this form. Instead, researchers are usually interested in evaluating the relative performance of stereopsis algorithms in particular regions of the input image – accuracy on occluded pixels (pixels whose projection is occluded in the other reference image), or near disparity discontinuities, for instance. To perform this sort of evaluation a greyscale mask image is created such that pixels to be excluded from the evaluation are assigned a greyscale value of zero. Figure 4.1 shows some example mask images from the well known Tsukuba image pair. Equation 4.1 modified to use a mask image $\mathcal{I}_{mask}$ is:



(a) Ground truth.

(b) All mask.

(c) Non-occluded region mask.

(d) Discontinuities mask.

Figure 4.1: Ground truth disparity map and evaluation masks for the left image of the Tsukuba image pair.

$$P_{mask} = \frac{\sum_{\mathbf{p}} T(\vec{\mathbf{c}}_{mask}(\mathbf{p}) \neq (0)) T(|\delta_c(\mathbf{p}) - \delta_t(\mathbf{p})| \geq \tau)}{\sum_{\mathbf{p}} T(\vec{\mathbf{c}}_{mask}(\mathbf{p}) \neq (0))} \tag{4.2}$$

where $\vec{\mathbf{c}}_{mask}(\mathbf{p})$ denotes the colour of pixel $\mathbf{p}$ in the mask image.

Some authors use evaluation metrics related to the percentage of incorrectly assigned pixels. For instance, Bhat and Nayar [5] simply report the number of pixels that are assigned incorrect disparity values, and Brockelbank and Yang [13] report the number of correctly matched features in their edge-matching stereopsis algorithm. Additionally, Birchfield and Tomasi [6] use line-plots of the disparity assignments for some scanlines in their calculated disparity maps that highlight which pixels are assigned incorrect disparity values.

Another common evaluation metric used by the stereopsis research community is the root mean square (RMS) error of the calculated disparity map – again, this metric is usually taken over a mask image to focus on areas of interest. The RMS error is calculated as:

$$RMS = \sqrt{\frac{\sum_{\mathbf{p}} T(\vec{\mathbf{c}}_{mask}(\mathbf{p}) \neq (0)) |\delta_c(\mathbf{p}) - \delta_t(\mathbf{p})|^2}{\sum_{\mathbf{p}} T(\vec{\mathbf{c}}_{mask}(\mathbf{p}) \neq (0))}} \tag{4.3}$$

where $\vec{\mathbf{c}}_{mask}(\mathbf{p})$ and $\delta_k(\mathbf{p})$ are defined as above.

When proposing a method for minimizing an energy equation for global stereopsis formulations some authors [3, 11, 32] have chosen to evaluate their method by presenting plots of the energy level of the best solution found versus time. Although this sort of evaluation is good for showing the relative strengths of different minimization methods, its value in the stereopsis domain is questionable in light of the works of Tappen and Freeman [90] and Barbu and Zhu [3] that show that the energy level of the ground truth disparity map is greater than that found during minimization.

Other metrics for quantitative evaluation that have been used by stereopsis researchers include: the false positive/negative rate in identifying certain classes of pixels [27], the execution speed of the algorithms being compared [33], and the density (percentage of the image that was assigned disparity values) of the calculated disparity map [39, 41].

4.1.3 Qualitative Evaluation

When evaluating stereopsis algorithms a qualitative evaluation is occasionally the only option. In the early days of stereopsis research any evaluation that utilized real photographs required a qualitative evaluation due to the lack of available technology for producing ground truth disparity maps. These days the technology exists (laser range finders, structured light reconstruction [83], et cetera) to create ground truth disparity maps for photographs taken under very controlled circumstances. That said, some authors [2, 3, 8, 14, 15, 17, 24, 62, 95, 107, 110] still choose to utilize a qualitative analysis on data sets for which ground truth is available.

4.2 Our Methodology For Evaluation

In this dissertation we analyze the performance of many different stereopsis algorithms. The goal is to create an analysis of the performance of these algorithms – from which we can draw conclusions about relative performance of the algorithms. To this end, we utilize a large data set consisting of 57 synthetic and photographic binocular stereo image pairs so as to provide large variety in the colours and shapes present in the data set. We also use a user-parameter optimization method [65] to reduce the probability that the results reflect a poor choice of user-parameters in the stereopsis algorithms rather than their expected performance. Finally, the relative performance of the stereopsis algorithms is compared using a ranking methodology that is designed to produce a statistically significant ranking of the compared algorithms. In this section we discuss each of these components of our analysis in turn.

4.2.1 Data Set

To perform our analysis we use a data set consisting of 30 synthetic and 27 photographic binocular stereo image pairs in the RGB colour space. The synthetic images were created using a freely available path-tracing ray tracer [75] with a custom plug-in written to capture the depth map of a rendered image along side the generated colour image. The use of these synthetic scenes allows us to vary the colour contrasts and textures used in the scenes to a greater degree than is available in the photographic data sets. The photographic image pairs used are the 2005 and 2006 $\frac{1}{3}^{rd}$-sized Middlebury stereo image data sets [46, 79]. From this data set we derive two additional data sets by introducing synthetic Gaussian sensor noise with two different variances; these additional data sets allow us to test algorithm performance when lower quality hardware is used for image capture.

Our 30 synthetic image pairs were created from three base geometric configurations consisting of basic geometric shapes and simple object models. Ten different random texture assignments were then applied to each of these base geometric configurations to obtain a total of 30 scenes for rendering. Figure A.1 shows the left images from the 30 synthetic image pairs in our data set. When referring to images from the synthetic portion of our data set we use the notation GgTn to specify an image pair. In this notation $g \in \{1, 2, 3\}$ specifies the geometric configuration that is used in the image pair, and $n \in \{1, 2, \ldots, 10\}$

distinguishes between the ten random texture assignments generated for that geometric configuration.

The 27 photographic image pairs from the Middlebury data sets that we use in this study are all staged scenes captured in a controlled environment to allow for the creation of the corresponding disparity maps using structured light reconstruction [83]. When capturing the images for the Middlebury 2005 and 2006 data sets, Scharstein and his colleagues captured each scene under all permutations of three lighting and three exposure settings; we use the image pairs captured under the second light and exposure settings – these settings result in images that are neither too dark nor washed out by too much lighting. Figure A.2 shows the left images from the 27 photographic images in our data set.

Introducing Synthetic Sensor Noise

It is common to synthetically simulate sensor noise by perturbing the colour values in the image with zero-mean Gaussian white noise; that is, for each pixel a perturbation value is generated according to a zero-mean Gaussian distribution and added to every colour channel of the pixel. However, Healey and Kondeputy [44] have observed that the variance of greyscale intensity noise for a CCD camera is dependent on the intensity of the light incident on the CCD sensor. Thus, although using zero-mean Gaussian white noise to synthesize noise in our synthetic images would be acceptable to the stereopsis research community, it is not realistic.

To develop a more realistic noise model we analyze the sensor noise characteristics of a Sony DCR-TRV230 NTSC digital camera by capturing 1022 images of a colourful static scene. By averaging all 1022 captured images, and then comparing the captured images against this mean image are able to calculate the per-channel sensor noise variance as a function of intensity as well as the noise correlation between colour channels. Figure 4.2 shows the noise variance for each colour channel by intensity, and the correlation matrix that was calculated in RGB colour space is:

$$\begin{pmatrix} 0.9999992803 & 0.6681645746 & 0.6844321738 \\ 0.6681645746 & 0.9999992835 & 0.6969747569 \\ 0.6844321738 & 0.6969747569 & 0.9999992738 \end{pmatrix} \tag{4.4}$$

To create noisy data sets we applied correlated Gaussian noise to each of the 30 synthetic image pairs and 27 photographic data pairs with two different multiples of the observed variance from the Sony DCR-TRV230 NTSC digital camera. The first level of noise used the observed mean and variance noise from the Sony camera, and the second used the observed mean with double the variance of the Sony camera. The result is three data sets; one with virtually no noise (the rendered synthetic images and the unmodified photographic images), one with noise roughly equivalent to those that would be captured by the Sony DCR-TRV230 NTSC digital camera, and one consisting of images roughly equivalent to those that would be captured by a poor quality digital camera (such as a webcam) or at a high ISO film setting.

4.2.2 User-Parameter Optimization

Virtually all stereopsis algorithms require tweaking of user-parameters to obtain decent results, and the algorithms used in this dissertation are no exception. Typically, practitioners and researchers manually tweak these parameters on a small set of training image pairs until they are satisfied with the results; a process that can be long and tedious for algorithms that do not execute in real-time. Additionally, it is not uncommon for user-parameter settings that produce good results on one image pair to produce less than stellar results on another – such as when one of the parameters is a gradient or truncation threshold. In this

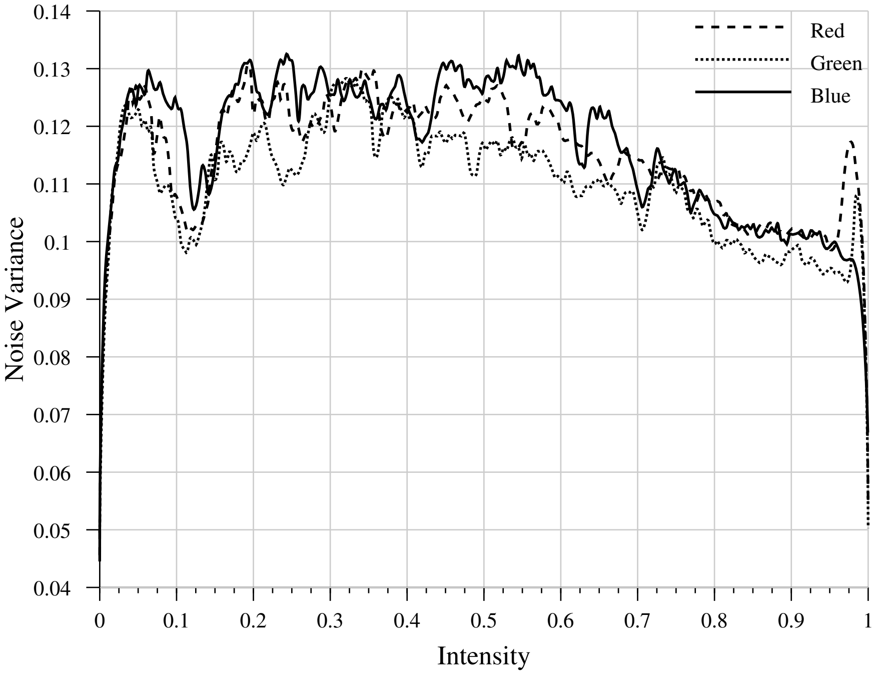


Figure 4.2: Sony DCR-TRV230 NTSC. Per-channel sensor noise variance as a function of intensity.

dissertation we use several hundred different stereopsis algorithms in a variety of settings, so an automated method for finding decent user-parameter settings is a necessity.

To optimize the user-parameters in this dissertation we adapt the response-surface optimization method proposed by Lizotte [65] to the stereopsis domain. Response-surface methods build a model of an objective function, f_{obj}, using all available data (function evaluations) about the objective, select a new point at which to evaluate f_{obj}, evaluate f_{obj} at the selected point, and incorporate the new data into the model. This process is then repeated as many times as desired. The method proposed by Lizotte uses a Gaussian process [77] to model the objective function, and provides a mechanism for identifying the sample point most likely to result in a better objective function value than has been observed thus far.

Since the method proposed by Lizotte is a black-box in nature, it does not require any knowledge of the source that is producing the values of f_{obj}; in fact, different objective functions can be "plugged in" to the method without making any changes to the implementation. Other parameter optimization methods that have been proposed for stereopsis [20, 79, 108] require that the stereopsis algorithm have a particular form, and require algorithm-specific derivation of the equations used for the parameter optimization; when optimizing user-parameters for hundreds of stereopsis algorithms, such constraints are unacceptable.

In this dissertation, we use the average percentage of incorrectly assigned pixels (Equation 4.2) between the left and right images in the image pair with $\tau = 1$ and a mask map that selects only those pixels in the ground truth with a valid (non-zero) disparity value for f_{obj}. We also tested several other objective functions – including the RMS error (Equation 4.3), and weighted sums of Equations 4.2 and 4.3 – but found that using Equation 4.2 as the objective function sufficient for our purposes.

In the remainder of this section we present each component of the parameter optimization method that we use in turn.

Gaussian Process Regression

Gaussian process regression [77] is a generalization of least-squares linear regression that allows for more flexible, non-linear, models and provides measures of confidence in the predictions made by the model. These models use a non-linear *kernel* function that describes the similarity of different inputs. These similarities in turn specify how the data define the shape of the resulting regression function.

The similarities are commonly defined in terms of the Euclidean distance between the input points, with a weight to adjust the importance of each axis. Given d-dimensional vectors $\vec{\mathbf{x}}$, and $\vec{\mathbf{z}}$, the kernel functions that we use in our work are the *squared exponential* (SE) kernel given by:

$$k_{\mathrm{SE}}(\vec{\mathbf{x}}, \vec{\mathbf{z}}) = \mathrm{e}^{-(\vec{\mathbf{x}}-\vec{\mathbf{z}})^T \mathbf{M} (\vec{\mathbf{x}}-\vec{\mathbf{z}})} \tag{4.5}$$

and the *rational quadratic* (RQ) kernel given by:

$$k_{\mathrm{RQ}}(\vec{\mathbf{x}}, \vec{\mathbf{z}}) = \left(1 + \frac{(\vec{\mathbf{x}}-\vec{\mathbf{z}})^T \mathbf{M} (\vec{\mathbf{x}}-\vec{\mathbf{z}})}{\beta}\right)^{-\beta}. \tag{4.6}$$

where $\mathbf{M} = \mathbf{D} + \mathbf{L}^T\mathbf{L}$ is a $d \times d$ covariance matrix with $\mathbf{D}$ being the $d \times d$ diagonal matrix:

$$D = \begin{pmatrix} \frac{1}{\ell_1^2} & 0 & 0 & \cdots & 0 \\ 0 & \frac{1}{\ell_2^2} & 0 & \cdots & 0 \\ 0 & 0 & \vdots & \ddots & 0 \\ 0 & 0 & 0 & \cdots & \frac{1}{\ell_d^2} \end{pmatrix} \tag{4.7}$$

and $\mathbf{L}$ the $m \times d$ whose rows define the m directions of high relevance.

Each of these two kernels has a length-scale parameter, ℓ_i, for each dimension that controls how smooth the regression function is along dimension i. The rational quadratic kernel has an additional parameter, β, that provides further control over the smoothness along all dimensions. In the limit as $\beta \to \infty$, k_{RQ} and k_{SE} are equivalent.

Given a set of observations, $X = \{f(\vec{\mathbf{x^i}}) = y_i : i = 1 \ldots n\}$, the posterior mean, $\bar{f}_*$, and posterior variance, $\mathbb{V}[f_*]$, of the Gaussian process with kernel $k(\vec{\mathbf{x}}, \vec{\mathbf{y}})$ are:

$$\bar{f}_*(\vec{\mathbf{x}}) = \vec{\mathbf{k}}^T(\sigma_f^2\mathbf{K} + \sigma_n^2\mathbf{I})^{-1}(\vec{\mathbf{y}} - \vec{\mathbf{u}}) + \mu \tag{4.8}$$

$$\mathbb{V}[f_*](\vec{\mathbf{x}}) = k(\vec{\mathbf{x}}, \vec{\mathbf{x}}) - \vec{\mathbf{k}}^T(\sigma_f^2\mathbf{K} + \sigma_n^2\mathbf{I})^{-1}\vec{\mathbf{k}} \tag{4.9}$$

where $\vec{\mathbf{k}}$ is the vector with $\vec{\mathbf{k}}_i = k(\vec{\mathbf{x^i}}, \vec{\mathbf{x}})$, $\mathbf{K}$ is the $n \times n$ matrix where the $(i,j)^{th}$ element is $k(\vec{\mathbf{x^i}}, \vec{\mathbf{x^j}})$, $\mathbf{I}$ is the $n \times n$ identity matrix, $\vec{\mathbf{y}}$ is the vector with $\vec{\mathbf{y}}_i = y_i$, μ is the prior mean of the data, $\vec{\mathbf{u}}$ is the vector with $\vec{\mathbf{u}}_i = \mu$, and σ_f^2 and σ_n^2 are the sample and signal variance of the data, respectively.

Notice that the Gaussian process itself contains a number of parameters that control how well it fits the data – μ, σ_f, and the kernel parameters ℓ_i, $\mathbf{L}_{i,j}$, and β. These parameters can be optimized to obtain a Gaussian process that better fits the data by maximizing the *log marginal likelihood* of the Gaussian process given its parameters. Given the same quantities as above, the log marginal likelihood, $\log p(\vec{\mathbf{y}}|X)$, of the Gaussian process is:

$$\log \rho(\vec{\mathbf{y}}|X) = -\frac{\vec{\mathbf{y}}^T(\sigma_f^2\mathbf{K} + \sigma_n^2\mathbf{I})^{-1}(\vec{\mathbf{y}} - \vec{\mathbf{u}}) + \log|\sigma_f^2\mathbf{K} + \sigma_n^2\mathbf{I}| + n\log(2\pi)}{2}. \tag{4.10}$$

The log marginal likelihood can be maximized by taking its partial derivatives with respect to each of the model parameters, and using any gradient ascent optimization method. We use the second implementation of the Broyden-Fletcher-Goldfarb-Shanno multidimensional minimizer (BFGS2) available in the GNU Scientific Library (GSL) version 1.11 to do this optimization.

Furthermore, a process known as *model selection* can be accomplished by finding the maximum log marginal likelihood of Gaussian processes with different kernel functions and choosing the model with the highest likelihood. Though one must be careful not to over fit the model to the data, this usually results in a better fit to the data.

In our implementation, we perform a model selection every ten iterations, and set the model parameters to maximize the log marginal likelihood in the remaining iterations. The model selection step is performed between the four kernels: SE and RQ with the matrix $\mathbf{L}$ in the decomposition of $\mathbf{M}$ omitted, and SE and RQ with $m = 1$.

At first glance, considering that the RQ kernel can approximate the SE kernel, it may seem odd to use both in our parameter optimization. Indeed, the RQ kernel is more flexible than the SE kernel, and will usually result in a better fit of the Gaussian process to the data. However, possible local maxima along the β axis while maximizing the log marginal likelihood of the model, and numerical instabilities caused by a very large β make inclusion of the SE kernel worth the minimal extra computational effort for added flexibility.

Initializing the Gaussian Process

When testing Lizotte's Gaussian process based surface-response parameter optimization method we observed that if the projection of the objective function to any of its domain dimensions is too "flat" then the matrix $\sigma_f^2\mathbf{K} + \sigma_n^2\mathbf{I}$ used in Equations 4.8, 4.9, and 4.10 can become ill-conditioned. This makes the process of finding its inverse numerically unstable, and results in the Gaussian process being unable to model the data.

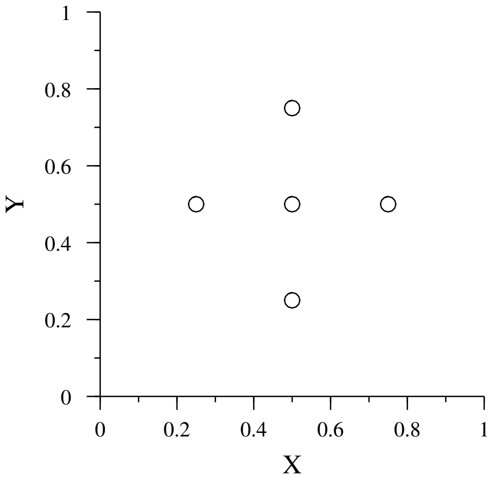


Figure 4.3: A plot of the five points in an axis-aligned star pattern on the domain $[0, 1] \times [0, 1]$.

To combat the problem of an ill-conditioned matrix, we initialize the Gaussian process using an axis-aligned star pattern in the domain of the objective function. That is, if the bounded domain of the objective function is given by $\prod_{i=1}^{n}[l_i, u_i]$, where l_i and u_i denote the lower and upper bounds of dimension i, respectively, then we evaluate the objective function at the $2n + 1$ points, $p_0, \ldots, p_{2n+1}$:

$$p_0 = (\frac{l_1+u_1}{2}, \frac{l_2+u_2}{2}, \ldots, \frac{l_n+u_n}{2}) \tag{4.11a}$$

$$p_{2k-1} = (\frac{l_1+u_1}{2}, \ldots, l_k + \frac{u_k-l_k}{4}, \ldots, \frac{l_n+u_n}{2}) \tag{4.11b}$$

$$p_{2k} = (\frac{l_1+u_1}{2}, \ldots, l_k + \frac{3(u_k-l_k)}{4}, \ldots, \frac{l_n+u_n}{2}) \tag{4.11c}$$

where k ranges from one to n. The objective function values at these $2n+1$ data points is then used to initialize the Gaussian process model. Figure 4.3 shows an example of one of these star patterns on the domain $[0,1] \times [0,1]$.

Selecting the Next Sample Point for Evaluation

Lizotte [65] suggests two methods for using the Gaussian process model of an objective function to select the next point at which to evaluate the objective function. Both of these methods are designed to select a point that is likely, based on the model, to result in an objective function evaluation that is better than the best observed value thus far.

Let G be a Gaussian process over the bounded domain Ω with posterior mean $f : \Omega \to \mathbb{R}$, and posterior variance $v : \Omega \to \mathbb{R}$ that is a model of the objective function $f_{\text{obj}} : \Omega \to \mathbb{R}$. Furthermore, let $X = \{\vec{\mathbf{x}^{\mathbf{i}}} : i = 1, \ldots, n\}$ be the n domain points at which f_{obj} was evaluated to construct G, $\vec{\mathbf{x}^{*}} = \arg\min_{\vec{\mathbf{x}} \in X} f_{\text{obj}}(\vec{\mathbf{x}})$, and $y^* = f_{\text{obj}}(\vec{\mathbf{x}^{*}}) - \eta$. The parameter η is discussed below.

Lizotte defines the probability of improvement (PI) for evaluating f_{obj} at some $\vec{\mathbf{x}} \in \Omega$ as:

$$\begin{aligned} \text{PI}(\vec{\mathbf{x}}) &= \rho(f_{\text{obj}}(\vec{\mathbf{x}}) < y^*) \\ &= \frac{y^* - f(\vec{\mathbf{x}})}{(v(\vec{\mathbf{x}}))^2} \end{aligned} \tag{4.12}$$

where $\rho(x)$ denotes the marginal probability of x.

He also defines the expected improvement (EI) of evaluating f_{obj} at some $\vec{\mathbf{x}} \in \Omega$ as:

$$\begin{aligned} \text{EI}(\vec{\mathbf{x}}) &= \mathbb{E}[y^* - \min\{y^*, f_{\text{obj}}(\vec{\mathbf{x}})\}] \\ &= (v(\vec{\mathbf{x}}))^2 \left(\text{PI}(\vec{\mathbf{x}})\Phi(\text{PI}(\vec{\mathbf{x}})) + \phi(\text{PI}(\vec{\mathbf{x}}))\right) \end{aligned} \tag{4.13}$$

where $\Phi(z) = \frac{1+\text{erf}(z)}{2}$, $\text{erf}(z) = \frac{2}{\sqrt{\pi}} \int_0^z e^{-t^2} dt$ is the error function, $\phi(z) = \frac{1}{\sqrt{2\pi}} e^{\frac{-z^2}{2}}$, and $\mathbb{E}[x]$ denotes the expected value of x.

The PI represents the chance that we will observe a $f_{\text{obj}}(\vec{\mathbf{x}})$ that is smaller than the best we have seen thus far, and the EI describes how much the reported smallest value, y^*, is expected to decrease once we obtain $f_{\text{obj}}(\vec{\mathbf{x}})$.

Lizotte suggests selecting the $\vec{\mathbf{x}}$ that maximizes either $\text{PI}(\vec{\mathbf{x}})$ (maximum probable improvement, or MPI) or $\text{EI}(\vec{\mathbf{x}})$ (maximum expected improvement, or MEI) as the next sample point to evaluate. The gradients of $\text{PI}(\vec{\mathbf{x}})$ and $\text{EI}(\vec{\mathbf{x}})$ are relatively easy to calculate, so any gradient ascent algorithm can be used to find such an $\vec{\mathbf{x}}$ – we use the BFGS2 multidimensional minimizer available in GSL version 1.11 with a log barrier [9] wrapper to transform the unbounded minimizer into a bounded minimizer.

The parameter η in the definition of y^* controls the degree of trade off between a low posterior mean and a high posterior variance in the point that maximizes $\text{PI}(\vec{\mathbf{x}})$ and $\text{EI}(\vec{\mathbf{x}})$. Low values of η result in a preference for points with a low posterior mean. In our implementation we use $\eta = \epsilon\sigma_f$ where σ_f is the sample variance of G and $\epsilon = 0.01$.

To select starting points for this maximization, we evaluate the EI (or PI) at 1000 jittered uniform random sample points in Ω and the n points in X, then use the 50 with the highest EI (or PI) as starting points for the maximization. The $\vec{\mathbf{x}}$ that results from the maximization with the highest EI (or PI) is then returned as the next sample point. However, if for each of the 50 converged-to points, $\vec{\mathbf{x}'}$, there exists an $\vec{\mathbf{x}} \in X$ such that $||\vec{\mathbf{x}} - \vec{\mathbf{x}'}|| < 0.01$ then we return a random point in Ω instead – this helps the point selection process avoid getting stuck at a locally optimum value.

We tested both selection methods for selecting user-parameters for stereopsis algorithms and found the point of maximum expected improvement (MEI) to yield slightly better, and more reliably consistent, results.

Stopping Criteria

The Gaussian process based response-surface parameter optimization method discussed thus far provides an iterative process for selecting user-parameters at which to evaluate the objective function. However, the question remains: when do we stop iterating? Two obvious stopping conditions are to stop either after a fixed number of iterations, or after a predefined time-limit has been reached. Another possibility is to try to detect when the process has converged; or, at least, does not seem to be finding any better objective function values than those found thus far.

In our experience, this response-surface method can iterate anywhere from a few times to several dozen times without finding any improvement in the objective function, and then suddenly find itself improving the objective for a few iterations. Due to this behaviour, automatically detecting convergence of the method proved too difficult.

Using a predefined time-limit has the benefit of always knowing that the process will never require more than the predefined amount of time. However, the algorithms we analyze in this dissertation require significantly varying amounts of execution time – anywhere from a few seconds to several minutes – so, using a predefined time-limit would put those algorithms that require more execution time at a disadvantage.

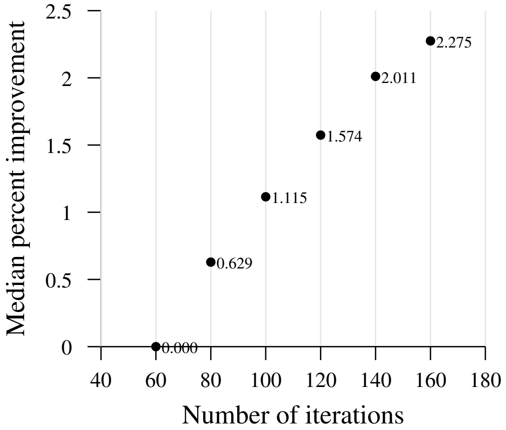


Figure 4.4: Number of point selection iterations versus median percent improvement over the 60 iteration results.

Thus, we stop iterating the user-parameter optimization once we have evaluated the objective function a fixed number of times. To determine a number of iterations that provides a good trade-off between user-parameter quality and execution time we conducted a small experiment where we use our optimization method, with the MEI point selection criteria, on 48 global stereopsis algorithms on the photographic portion of our no-noise data set at five different iteration thresholds. The 48 algorithms we use are derived from combining the SO1, SO2, HBPTL, and HBPG global stereopsis frameworks (see Section 5.2) with the SqAD, SqGD, SqSD, Sq[t]AD, Sq[t]GD, Sq[t]SD, SwAD, SwGD, SwSD, Sw[t]AD, Sw[t]GD, Sw[t]SD constructable match cost functions (see Section 2.2.2). Figure 4.4 shows a graph of the median relative improvement obtained for each each iteration count over the results obtained with 60 iterations.

Based on these results and the sheer number of times our optimization method is used to obtain the results in this dissertation we decided to use 100 sample points. We find this number of sample points to yield a good trade-off between user-parameter quality and execution time; we generally obtain parameter values on par in quality with what we would expect to obtain with manual parameter tweaking.

Furthermore, using a constant number of iterations ensures that the user-parameter optimizer provides a level playing field for our analysis by performing the same amount of work on each stereopsis algorithm.

4.2.3 Cluster Ranking

The typical approach (for example, [80]) when quantitatively analyzing stereopsis algorithms is to collect a metric of performance for each algorithm from a single image in each binocular image pair in the testing data set. Once these metrics have been collected they are sorted to achieve a ranking of the algorithms on each image pair, and these rankings are averaged to obtain an overall ranking of the algorithms being compared. However, we find this approach lacking in several respects:

1. It uses performance metrics from only half of the images available in the test data set; and
2. It does not implement a mechanism for identifying when the results generated by two, or more, algorithms cannot be statistically differentiated.

To address these problems we devised a simple greedy ranking algorithm [69] based on two statistical significance tests: single-factor between-subjects analysis of variance [84] (ANOVA), and the Friedman two-way analysis of variance by ranks [84] (the Friedman test). ANOVA [84] is the main method employed by the medical community in their clinical trials to determine whether or not a given treatment has any effect. Given $k \geq 2$ groups of normally distributed sample data, ANOVA gives the probability that at all of the groups represent populations with the same mean value. The Friedman test is similar to ANOVA except that it uses ordinal data instead of normally distributed data. Given k "tests", the result of which can be quantified with an ordinal value, and N "subjects" who each perform all k tests, the Friedman test gives the probability that at all of the k tests represent populations with the same median value.

Our algorithm produces both an overall and per-image-pair ranking of the stereopsis algorithms under consideration such that any two stereopsis algorithms are assigned different ranks if they produce statistically dissimilar results at some user-defined confidence threshold, C. The overall ranking is calculated in two phases. In the first phase, a ranking of all of the stereopsis algorithms is calculated for each of the image pairs in the test data set. The second phase then uses these rankings to calculate an overall ranking. The algorithm used to calculate the rankings in both phases is structurally identical; the only differences being the data (quantitative error metrics versus ordinal rank data) and statistical test used to calculate the ranking (ANOVA in phase one, and the Friedman test in phase two).

Given a confidence level C, and N algorithms to rank, the cluster ranking algorithm calculates a ranking as follows:

1. Sort the algorithms in descending order of performance. Store result in the list L.
   - In phase one: Since ANOVA tests for statistically similar group means, this sort is performed using the average performance metric between the two images in the image pair as the sort key.
   - In phase two: Since the Friedman test tests for algorithms with statistically similar median rank, this sort is performed using the median rank of the algorithm over all image pairs in the test data set. Ties are broken such that lower mean ranks appear earlier in the list.
2. Partition L such that algorithms in the same partition are statistically similar at confidence level C:
   (a) Note that if there is only one data point for each element of L, then this partitioning is performed by assigning tied values to the same partition.
   (b) Assign the first element of L to the first partition.
   (c) Consecutively add elements of L as long as all of the algorithms in the partition are statistically similar at confidence C.
      - In phase one: The ANOVA test is used to determine similarity with the sample data for each algorithm being the performance metric from both images in the image pair. We expect that the performance metric for an algorithm on the left and right images in the image pair are sufficiently similar to satisfy the normal distribution requirement of the ANOVA test.
      - In phase two: The Friedman test is used to determine similarity with the sample data for each algorithm being the list of the algorithm's ranks on all image pairs in the test data set. That is, the algorithms correspond with the "tests" and the image pairs with the "subjects" in the above description of the Friedman test.

   (d) If all elements of L have been assigned to a partition, then end.
   (e) Else, start a new partition with the next element of L and goto (b).
3. Assign each algorithm a rank equal to the rank order of its assigned partition in L.

4.3 Summary

There are many different ways to evaluate the accuracy of computational stereopsis algorithms – ranging from quantitative to qualitative. Ideally, an evaluation methodology will not simply suggest that algorithm X is better than algorithm Y, but will also allow a researcher to hypothesize as to why that is the case.

In the absence of a standardized methodology and data set for evaluating stereopsis algorithms, comparing algorithms from different sources can be a difficult and time-consuming process since the algorithms will have to be reimplemented, and appropriate user-parameters found for comparison. For this purpose Scharstein and Szeliski [82] introduce an evaluation methodology, and data set, that has become the de facto standard for comparative evaluation of stereopsis algorithms. However, the introduced methodology uses a very small data set (four image pairs), does not address the problem of statistical significance of the results, and can be affected by the amount of time researchers are willing to spend tweaking the user-parameters of their algorithms for optimal results.

To perform a comparative evaluation of the stereopsis algorithms investigated in this dissertation we present an evaluation methodology that uses a large data set, automatically optimizes user-parameters in such a way as to ensure a level field for analysis, and provides a ranking where algorithms that are given the same rank produce statistically similar results. Our data set consists of synthetic and photographic binocular stereo image pairs with various amounts of synthetically added sensor noise. To optimize user-parameters in the stereopsis algorithms being compared we adopt the Gaussian process based response-surface optimization method proposed by Lizotte [65] to the stereopsis domain. Finally, we propose a simple greedy ranking algorithm that uses statistical significance testing to identify when two, or more, stereopsis algorithms are producing statistically similar results.

Chapter 5

Match Cost Functions in Global Stereopsis

The first goal in this dissertation is to begin investigating the question: What makes a match cost function good for global stereopsis algorithms? There are hundreds, if not thousands, of different match cost functions available for use in global stereopsis algorithms, but just what makes a match cost function good is not well understood. In local stereopsis algorithms a match cost function whose minimum corresponds with the correct disparity value is often sufficient. In global algorithms there is an interaction between the match cost function and the smoothing assumptions that must be considered.

In this dissertation we begin on the long road to understanding what makes a match cost function good for global stereopsis algorithms with a large scale study that compares the relative performance of match cost functions, and their component parts, in the context of global stereopsis. From the results of this study we are able to infer a set of guidelines that we believe can help find good match cost functions for global stereopsis.

For our study we use the constructable match cost functions described in Section 2.2.2 within five different global stereopsis frameworks. Our use of constructable match cost functions allows us to not just analyze the relative performance of the functions, but to systematically analyze the relative performance of the components of the functions in such a way as to gain some insight into what makes a good match cost function. The five global stereopsis frameworks we use in this investigation are a representative sampling of the types of algorithms that are used by the research community; they include three energy formulations of stereopsis (with one being a formulation over an image segmentation instead of over image pixels), and two probability formulations.

We split our study of match cost functions for global stereopsis algorithms into two parts: match cost functions that do not include a spatial aggregate, and those that do. The primary reason for this split is the shear computational requirements of spatially aggregated match cost functions if we did not make the split; many spatial aggregates have substantial execution time requirements. In the first part of our study – for functions with no spatial aggregate – we consider a total of 144 match cost functions. Were we to consider the application of all five spatial aggregates to each of these 144 match cost functions the computational requirements of the resulting 720 match cost functions would be overwhelming, and likely unnecessary to gain useful insight on spatial aggregates. Instead, we study the application of spatial aggregates to four of the better performing match cost functions from the first phase; resulting in 20 match cost functions to consider.

The data for this study was collected by running our stereopsis program on the Glacier [99] and Terminus [101] clusters of the Western Canada Research Grid (WestGrid). The Glacier cluster consists of 840 IBM eServer BladeCenter HS20 computers with dual 3.0GHz Xeon processors, running a version of the RedHat Linux operating system, configured as a Beowulf cluster. The Terminus cluster is comprised of 468 BL465c G1 CTO Blade computers with dual-core 2.4 GHz AMD Opteron processors, running a version of the Linux operating system, and is also configured as a Beowulf cluster. Gathering the data for the first phase of this study was conducted entirely on the Glacier cluster, and required approximately 6.159 years of CPU time. The data collection for the second phase was conducted primarily on the Glacier cluster with some collection performed with the Terminus cluster; the execution speed difference of our program on these two clusters is negligible. In total, the data collection for the second phase of our study required approximately 20.636 years of CPU time to complete.

We begin this Chapter with a brief discussion of related work, and then present the five stereopsis frameworks we use in our investigation. We then move on to discuss the results of our study; beginning with the match cost functions that do not include a spatial aggregate.

5.1 Related Work

Some authors [5, 6, 30, 46, 54, 91, 92] have conducted small scale comparisons of match cost functions for stereopsis. These comparative studies are small scale in that they compare relatively few match cost functions, they generally do so within only one stereopsis algorithm (typically winner-take-all), and they generally use data sets that are small. In this section, we outline some of the studies made by these authors.

Bhat et al. [5] compare four ordinal match cost functions, that are based on the relative ordering of greyscale intensity values within a window, and the greyscale sum of absolute differences function over a window with the winner-take-all algorithm. Birchfield and Tomasi [6] show that their proposed greyscale match cost function performs better than the L1 norm metric using a dynamic programming algorithm that performs the optimization along horizontal scan lines. El-Mahassni [30] compares five greyscale match cost functions using the winner-take-all algorithm. Khaleghi et al. [54] compare five distance metrics (Manhattan, Euclidean, Canberra, squared chord, and square chi-squared) as greyscale match cost functions with the winner-take-all algorithm. Tombari et al. [92] compare their proposed match cost function with an adaptive support weight [105] match cost function using the winner-take-all algorithm.

The closest in scale that any previous studies have come to our study on the effects of match cost function choice in global stereopsis algorithms are Hirschmüller and Scharstein [46], who do utilize global stereopsis algorithms in their study, and Tombari et al. [91], who only compare match cost functions using winner-take-all. Hirschmüller and Scharstein [46] compare six greyscale match cost functions using a winner-take-all based local algorithm, a global algorithm using graph cuts, and a global algorithm using dynamic programming. The goal in their evaluation is to analyze the tolerance of these match cost functions to radiometric differences, due to differences in exposure time and/or lighting, between the two images in an image pair. Tombari et al. [91] compare 14 spatial aggregates for match cost functions with the winner-take-all algorithm using the Middlebury stereo evaluation framework [80]. Their evaluation applies the spatial aggregates to a single match cost function (the SAD match cost function) and addresses both processing speed and accuracy of the match cost functions compared.

To our knowledge no other study has attempted to analyze match cost functions on a component level, or at the scale that we do in this dissertation.

5.2 Global Stereopsis Frameworks

For our analysis of the performance of match cost functions in the context of global stereopsis algorithms we use five different algorithmic frameworks into which we "plug in" each match cost function. This set of relatively simple frameworks was chosen to form a representative sampling of the types of global stereopsis algorithms that have been proposed over the years – subject to the constraint that the resulting stereopsis algorithms must have a relatively short execution time.

Each of the stereopsis algorithms considered during our study are executed approximately 17,000 times. Thus, even with the use of substantial computing resources, short execution times are pivotal to ensuring data collection does not require an unreasonable amount of time. For this reason frameworks that utilize slower optimization methods – simulated annealing, graph cuts, et cetera – are not included.

We refer to the five global stereopsis frameworks that we use in our analysis as: SO1, SO2, HBPTL, HBPG, and RTree. The SO1 and SO2 frameworks are based on an energy minimization formulation of stereopsis defined over image pixels, with their solutions found through dynamic programming optimization. The RTree framework is also based on an energy formulation of stereopsis and is optimized via dynamic programming on a structure called the region tree [63]. However, this energy equation is defined over segments derived from a segmentation of the image instead of directly on image pixels. The HBPG and HBPTL frameworks are both based on a probability formulation of stereopsis and utilize hierarchical loopy belief propagation [32] to find a maximum-a-priori solution of this probability function.

Parameter	Range	Parameter	Range
α	$[0,1]$	β	$[1,100]$
P_{tr}	$[10^{-6},0.5]$	c	$[1,5]$
σ_s	$[2,15]$	ϵ	$[10^{-6},1]$
λ	$[10^{-6},1]$	c_r	$[1,7.5]$
s	$[10^{-6},10]$	d	$[10^{-6},1]$

Table 5.1: Ranges used for stereopsis framework user-parameters during user-parameter optimization.

Each of these stereopsis frameworks contains a number of user-defined parameters that require optimization. We use our response-surface based optimization method (see Section 4.2.2) for this, and bound the parameter values to the ranges given in Table 5.1.

For the remainder of this section, let $\mathcal{I}_s$ denote the source image that we are interested in calculating a disparity map for, and $\mathcal{I}_m$ denote the other image in the given image pair. Let $c(\mathbf{p},\delta)$ denote a match cost function, and $c_{\max} = \max\{c(\mathbf{p},\delta) : \mathbf{p} \in \mathcal{I}_s, \delta \in \{1,\ldots,n\}\}$ where n is the maximum disparity value for the image pair under consideration (n is equal to the baseline separation between the left and right images in the image pair).

5.2.1 The SO1 and SO2 Frameworks

The SO1 and SO2 – shorthand for *scan line optimization #1* and *#2* – frameworks both utilize an energy minimization formulation of stereopsis. They differ in the optimization technique and smoothing term used. Both of these frameworks calculate a disparity map, f_s, for $\mathcal{I}_s$ that is an approximate minimum of the energy equation:

$$E(f_s) = \sum_{\mathbf{p}\in\mathcal{I}_s} \frac{c(\mathbf{p}, f_s(\mathbf{p}))}{c_{\max}} + \sum_{\{\mathbf{p},\mathbf{q}\}\in\mathcal{N}_s} V_{\mathbf{p},\mathbf{q}}(f_s(\mathbf{p}), f_s(\mathbf{q})) \tag{5.1}$$

where $\mathcal{N}_s$ a set of neighbour pixel pairs in the image $\mathcal{I}_s$, and $V_{\mathbf{p},\mathbf{q}}(\delta_\mathbf{p}, \delta_\mathbf{q})$ is a smoothing function. $\mathcal{N}_s$ contains all four-connected neighbour pixel pairs in the SO1 framework, and all eight-connected neighbour pixel pairs in the SO2 framework.

The SO1 framework is derived from the algorithm proposed by Mattoccia et al. [67]. It applies scan line optimization (a form of dynamic programming) over horizontal and vertical scan lines in $\mathcal{I}_s$ to find f_s. This framework uses the smoothing function:

$$V_{\mathbf{p},\mathbf{q}}(\delta_\mathbf{p}, \delta_\mathbf{q}) = \begin{cases} 0 & \text{if } |\delta_\mathbf{p} - \delta_\mathbf{q}| = 0 \\ \Lambda_\alpha(\mathbf{p}, \delta_\mathbf{p}, \mathbf{q}, \delta_\mathbf{q}) & \text{if } |\delta_\mathbf{p} - \delta_\mathbf{q}| = 1 \\ \Lambda_{\alpha\beta}(\mathbf{p}, \delta_\mathbf{p}, \mathbf{q}, \delta_\mathbf{q}) & \text{otherwise} \end{cases} \tag{5.2}$$

where $\alpha > 0$ and $\beta > 1$ are user-defined parameters, and the function $\Lambda_\eta(\mathbf{p}, \delta_\mathbf{p}, \mathbf{q}, \delta_\mathbf{q})$ is defined as:

$$\Lambda_\eta(\mathbf{p}, \delta_\mathbf{p}, \mathbf{q}, \delta_\mathbf{q}) = \begin{cases} \eta & \text{if } ||\vec{\mathbf{c}}(\mathbf{p}) - \vec{\mathbf{c}}(\mathbf{q})|| < P_\text{tr} \text{ and } ||\vec{\mathbf{c}}(\mathbf{p}_{\delta_\mathbf{p}}) - \vec{\mathbf{c}}(\mathbf{q}_{\delta_\mathbf{q}})|| < P_\text{tr} \\ \frac{\eta}{2} & \text{if } ||\vec{\mathbf{c}}(\mathbf{p}) - \vec{\mathbf{c}}(\mathbf{q})|| \geq P_\text{tr} \text{ and } ||\vec{\mathbf{c}}(\mathbf{p}_{\delta_\mathbf{p}}) - \vec{\mathbf{c}}(\mathbf{q}_{\delta_\mathbf{q}})|| < P_\text{tr} \\ \frac{\eta}{2} & \text{if } ||\vec{\mathbf{c}}(\mathbf{p}) - \vec{\mathbf{c}}(\mathbf{q})|| < P_\text{tr} \text{ and } ||\vec{\mathbf{c}}(\mathbf{p}_{\delta_\mathbf{p}}) - \vec{\mathbf{c}}(\mathbf{q}_{\delta_\mathbf{q}})|| \geq P_\text{tr} \\ \frac{\eta}{4} & \text{if } ||\vec{\mathbf{c}}(\mathbf{p}) - \vec{\mathbf{c}}(\mathbf{q})|| \geq P_\text{tr} \text{ and } ||\vec{\mathbf{c}}(\mathbf{p}_{\delta_\mathbf{p}}) - \vec{\mathbf{c}}(\mathbf{q}_{\delta_\mathbf{q}})|| \geq P_\text{tr} \end{cases} \tag{5.3}$$

where $\mathbf{p}_\delta$ and $\mathbf{q}_\delta$ denote the warps of $\mathbf{p}$ and $\mathbf{q}$ from $\mathcal{I}_s$ to $\mathcal{I}_m$ at disparity δ, respectively, and $P_\text{tr} > 0$ is a user-defined threshold parameter.

The SO2 framework is derived from the algorithm proposed by Hirschmüller [45]. Rather than performing a scan line optimization over only the horizontal and vertical scan lines of $\mathcal{I}_s$, as in the SO1 framework, this framework employs a 16-directional scan line optimization. In this framework, the smoothing function is defined as:

$$V_{\mathbf{p},\mathbf{q}}(\delta_\mathbf{p}, \delta_\mathbf{q}) = \begin{cases} 0 & \text{if } |\delta_\mathbf{p} - \delta_\mathbf{q}| = 0 \\ P_1 & \text{if } |\delta_\mathbf{p} - \delta_\mathbf{q}| = 1 \\ \max\{P_1, \frac{P_1 P_2 \max\{||\vec{\mathbf{c}}(\mathbf{p}') - \vec{\mathbf{c}}(\mathbf{q}')|| : \{\mathbf{p}', \mathbf{q}'\} \in \mathcal{N}_s\}}{||\vec{\mathbf{c}}(\mathbf{p}) - \vec{\mathbf{c}}(\mathbf{q})||} & \text{otherwise} \end{cases} \tag{5.4}$$

where $P_1 > 0$ and $P_2 \geq 1$ are user-defined parameters, and $\mathcal{N}_s$ is the set of all eight-connected neighbour pixel pairs in $\mathcal{I}_s$.

5.2.2 The RTree Framework

The RTree – shorthand for *region tree* – stereopsis framework is based on the region tree method proposed by Lei et al. [63]. Our region tree method proceeds as follows. First, a mean-shift segmentation algorithm [23], in the CIELuv colour space, with a spatial bandwidth of two, a minimum segment size of 200 pixels, and a user-defined spectral bandwidth, $\sigma_s > 0$, is used to calculate a segmentation, $S(\mathcal{I}_s)$, of $\mathcal{I}_s$. This segmentation is used to find a disparity assignment, g, that assigns a disparity value to each segment in the obtained segmentation, via dynamic programming on a region tree; all pixels within a segment are assigned the disparity value assigned to the segment. Specifically, we aim to find a g that is an approximate minimum of:

$$E(g) = \sum_{s \in S(\mathcal{I}_s)} \sum_{\mathbf{p} \in s} \frac{c(\mathbf{p}, g(s))}{c_\text{max}} + \sum_{\{s_1, s_2\} \in S'(\mathcal{I}_s)} V_{s_1,s_2}(g(s_1), g(s_2)) \tag{5.5}$$

where $S'(\mathcal{I}_s)$ is the set of four-connected adjacent segment pairs in $S(\mathcal{I}_s)$, and $V_{s_1,s_2}(a, b)$ is defined as:

$$V_{s_1,s_2}(a,b) = l(s_1,s_2)\begin{cases} 0 & \text{if } |a-b| = 0 \\ c_r\lambda & \text{if } |a-b| = 1 \text{ and } ||\bar{\vec{\mathbf{c}}}(s_1) - \bar{\vec{\mathbf{c}}}(s_2)|| < \epsilon \\ \lambda & \text{otherwise} \end{cases} \tag{5.6}$$

where $l(s_1, s_2)$ is the edge length between segments s_1 and s_2, $\bar{\vec{\mathbf{c}}}(s)$ denotes the average colour of over all pixels in segment s, and $c_r > 1$, $\lambda > 0$, and $\epsilon > 0$ are user-defined parameters.

Once the segmentation has been obtained, we construct a weighted segment-adjacency graph, G, from the segmentation such that the weight of an edge between two vertices is the Euclidean distance between the average colours of their corresponding segments. From G we construct a minimum spanning tree, M, and set the edge weight of an edge, $e = (u, v)$, in M to the value of $V(u, v)$. Finally, the tree-based dynamic programming method described by Lei et al. is applied to M to arrive at an approximate solution to Equation 5.5.

5.2.3 The HBPTL and HBPG Frameworks

The HBPTL and HBPG – shorthand for *hierarchical belief propagation* with either a *truncated linear* (HBPTL) or *gradient-based* (HBPG) prior – frameworks are both modifications of Felzenswalb and Huttenlocher's [32] proposed stereopsis algorithm. They propose using hierarchical loopy belief propagation (HBP) to find an approximate solution to an energy equation formulation of stereopsis. We have modified their proposed HBP method to find an approximate maximum-a-priori (MAP) solution, f, to the following probability formulation of stereopsis:

$$p(f|\mathcal{I}_s, \mathcal{I}_m) \propto \prod_{\mathbf{p}\in\mathcal{I}_s} \rho_{\mathbf{p}}(f(\mathbf{p})) \prod_{\{\mathbf{p},\mathbf{q}\}\in\mathcal{N}_s} \rho_{\mathbf{p},\mathbf{q}}(f(\mathbf{p}), f(\mathbf{q})) \tag{5.7}$$

where f is the disparity map for the source image, $\mathcal{N}_s$ is the set of all four-connected neighbour pixel pairs in $\mathcal{I}_s$, $\rho_{\mathbf{p},\mathbf{q}}(\delta_{\mathbf{p}}, \delta_{\mathbf{q}})$ denotes the smoothing prior, and the data term, $\rho_{\mathbf{p}}(\delta)$, is defined as:

$$\rho_{\mathbf{p}}(\delta) = e^{-c(\mathbf{p},\delta)/c_{\max}}. \tag{5.8}$$

In the HBPTL framework, we use the truncated linear prior with user-defined parameters $s, d > 0$:

$$\rho_{\mathbf{p},\mathbf{q}}(\delta_{\mathbf{p}}, \delta_{\mathbf{q}}) = e^{-\min\{s|\delta_{\mathbf{p}}-\delta_{\mathbf{q}}|, d\}} \tag{5.9}$$

The HBPG framework uses the gradient-based prior:

$$\rho_{\mathbf{p},\mathbf{q}}(\delta_{\mathbf{p}}, \delta_{\mathbf{q}}) = \begin{cases} 1 & \text{if } \delta_{\mathbf{p}} = \delta_{\mathbf{q}} \\ e^{-c\lambda} & \text{if } \frac{||\vec{\mathbf{c}}(\mathbf{p}) - \vec{\mathbf{c}}(\mathbf{q})||}{\max\{||\vec{\mathbf{c}}(\mathbf{p}') - \vec{\mathbf{c}}(\mathbf{q}')|| : \{\mathbf{p}',\mathbf{q}'\}\in\mathcal{N}_s\}} < \epsilon \text{ and } \delta_{\mathbf{p}} \neq \delta_{\mathbf{q}} \\ e^{-\lambda} & \text{otherwise} \end{cases}. \tag{5.10}$$

In our implementation of the HBPTL and HBPG frameworks we use three hierarchy levels, and perform 40 belief propagation steps per level.

5.3 Match Cost Functions With No Spatial Aggregate

In the first phase of our study on match cost functions for global stereopsis algorithms we consider a set of constructable match cost functions (see Section 2.2.2) that do not include a spatial aggregate in their definition. This set is constructed from all combinations of the match cost function components listed in Tables 2.1a, 2.1b, and 2.1c along with the truncated versions of all of the channel norm functions, and channel aggregates except for the *max* channel aggregate. Since it is a simple max operator, truncating the result of the *max* channel aggregate is unnecessary when truncating channel norm functions. In total, the set of match cost functions for this phase of our study contains 144 elements.

Rather than manually choosing what user-defined parameter values to use in each of these match cost functions, we allow our user-parameter optimization method (see Section 4.2.2) to optimize these parameters. It is possible that a selection of user-parameter values for a match cost function component will work well within the context of one match cost function, but not work as well within another. It is also possible that a set of user-parameters for a match cost function work well within one, or more, of our stereopsis frameworks, but not as well in the others. Furthermore, letting our user-parameter optimizer handle the selection of the match cost function user-parameters allows us to analyze the distributions of optimized parameter values. It also allows us to focus on a more high-level aggregate analysis on the performance of forms of match cost functions without having to worry about whether the chosen user-parameters are good in all circumstances.

Function Name	**Parameter range(s)**
Generalized exponential	$s \in [10^{-6}, 3]$
Lorentzian p-function	$\sigma \in [10^{-6}, 3]$
Truncation	$t \in [1/255, 50/255]$

(a) Channel norm function parameter ranges.

Function Name	**Parameter range(s)**
Weighted sum	$u_1, u_2 \in [0, 1]$
Truncation	$t \in [1/255, 50/255]$

(b) Channel aggregate parameter ranges.

Table 5.2: User-parameter ranges for channel norm function, and channel aggregate constructable match cost function components.

In Tables 5.2a and 5.2b we list the ranges used by our optimizer for each of the match cost function parameters used in this study. These parameter ranges were chosen based on some preliminary experimentation we performed with each of the match cost function components. Recall that we divide the match cost function by $c_{\max}$ in each of our stereopsis frameworks. One result of this is that the σ parameter of the generalized exponential channel norm function can be canceled out of the equation. Thus, we set the value of σ to one when using a generalized exponential channel norm function. Furthermore, we restrict the values of the weight parameters in the weighted sum channel aggregate such that they sum to unity. Since this sort of constraint cannot be directly implemented in our parameter optimization technique we optimize parameters u_1 and u_2 for the weighted sum instead, and derive the weights w_1, w_2, and w_3 from those values as:

$$w_1 = u_1 \tag{5.11}$$

$$w_2 = u_2(1 - u_1) \tag{5.12}$$

$$w_3 = 1 - w_1 - w_2. \tag{5.13}$$

To evaluate the accuracy of each disparity map generated by a global stereopsis algorithm in this study we calculate the percentage of pixels that differ by more than one from the given ground truth disparity over a mask image; see Equation 4.2. The mask image that we use for this evaluation contains zero-valued pixels only where the ground truth disparity map has no valid disparity assigned and in a small border on either the left or right side of the image. This border is equal in width to the baseline separation between the two images in the image pair, is on the left side in the left image, and the right side in the right image.

Recall that our full data set consists of three separate data sets, each with different amounts of synthetic sensor noise – no noise introduced, small variance-magnitude sensor noise, and large variance-magnitude sensor noise. For this analysis we consider the relative rankings on each of these data sets independently. We calculate rankings for each of our data sets using cluster ranking at 95% confidence as follows. For each global stereopsis framework (SO1, SO2, et cetera) we first calculate a ranking of all match cost functions for each of the 57 stereo image pairs in the data set. We then use cluster ranking on these ordinal ranks to obtain per-framework and over all rankings of the match cost functions, function forms, or components we are interested in studying.

5.3.1 Truncation Form Analysis

There are three possible ways of adding truncation to a given a constructable match cost function, XYZ, with channel aggregate X, channel norm function Y, and channel function Z. We can truncate the channel norm function (resulting in the function XY[t]Z), we can truncate the channel aggregate (resulting in the function X[t]YZ), or we can truncate both (resulting in the function X[t]Y[t2]Z).

The method typically employed by the stereopsis research community is to truncate the result of the channel aggregate; that is, use the function X[t]YZ. Is this method of truncation ideal or just a throwback to the early days of stereopsis algorithm research where only greyscale images were used? To our knowledge, no one has thought to ask this question. In this section, we analyze the data collected during the first phase of our match cost function study to seek an answer to this question.

For the aggregate, over all, portion of this analysis we consider all of the data generated in our study by using match cost functions that do not include the *max* channel aggregate. When using the max channel aggregate X[t]YZ and XY[t]Z are equivalent, and X[t]Y[t2]Z is redundant. So, it does not make sense to include the match cost functions that use this channel aggregate in this part of the analysis.

The data in Table 5.3a indicates that when there is no sensor noise the common method of truncating a match cost function (form X[t]YZ) can generally be expected to produce statistically significantly better results over all. However, the data also indicates that we can expect simply truncating the match cost function in some manner to be sufficient to improve accuracy of the resulting disparity maps for most global stereopsis frameworks.

The data in Tables 5.3b and 5.3c indicate that as the magnitude of sensor noise is increased the results that can be expected from the XY[t]Z form gradually improve versus those of the X[t]YZ. When there is high magnitude sensor noise present in the input images we can expect statistically better results by truncating only the channel norm function of a match cost function. We expect that truncating the channel norm function of a match cost function limits the effect of large magnitude noise on the reported match quality between pixels more effectively than truncating the results of the channel aggregate.

As usual, there are exceptions to this. The data also indicates that performing no truncation in the match cost function produces results with no statistically significant difference from the XY[t]Z truncation form in the SO1 and SO2 frameworks on input with high magnitude noise. It also suggests that simply truncating the match cost function in some manner is sufficient to improve accuracy of stereopsis algorithms based on the RTree, HBPTL, and HBPG frameworks.

In general, our data indicates that truncating a match cost function in some manner produces statistically better results than not truncating the match cost function at all.

The over all, and per-framework rankings of the truncation forms for each match cost function are listed in the remaining tables of Appendix B. Although this data does generally

Truncation form	Over all	SO1	SO2	RTree	HBPTL	HBPG
XYZ	3	2	2	2	2	3
XY[t]Z	2	1	1	1	1	2
X[t]YZ	1	1	1	1	1	1
X[t]Y[t2]Z	2	1	1	1	1	1

(a) Rankings for data set with no noise introduced.

Truncation form	Over all	SO1	SO2	RTree	HBPTL	HBPG
XYZ	3	1	2	2	2	2
XY[t]Z	1	1	1	1	1	1
X[t]YZ	1	1	3	1	1	1
X[t]Y[t2]Z	2	1	3	1	1	1

(b) Rankings for data set with lowest magnitude noise introduced.

Truncation form	Over all	SO1	SO2	RTree	HBPTL	HBPG
XYZ	2	1	1	2	2	2
XY[t]Z	1	1	1	1	1	1
X[t]YZ	2	2	2	1	1	1
X[t]Y[t2]Z	3	3	3	1	1	1

(c) Rankings for data set with highest magnitude noise introduced.

Table 5.3: Cluster rankings of truncation forms of match cost functions at 95% confidence. Rankings displayed are the over all rankings, and the overall rankings for each global stereopsis framework.

conform with our analysis of the relative performance of the general truncation forms, it also shows that the manner in which to truncate a given match cost function within a given stereopsis framework should be investigated by the researcher/developer to ensure the best possible results.

5.3.2 Channel Function Analysis

Channel function	Over all	SO1	SO2	RTree	HBPTL	HBPG
B	2	2	2	2	2	1
D	1	1	1	1	1	1

(a) Rankings for data set with no noise introduced.

Channel function	Over all	SO1	SO2	RTree	HBPTL	HBPG
B	2	2	2	1	2	1
D	1	1	1	1	1	1

(b) Rankings for data set with lowest magnitude noise introduced.

Channel function	Over all	SO1	SO2	RTree	HBPTL	HBPG
B	1	2	1	1	1	1
D	1	1	2	1	1	1

(c) Rankings for data set with highest magnitude noise introduced.

Table 5.4: Cluster rankings of channel functions at 95% confidence. Rankings displayed are the over all rankings, and the overall rankings for each global stereopsis framework.

In this study we utilize two different channel functions when constructing our match cost functions: the difference function (D), and the greyscale Birchfield and Tomasi measure (B).

Noise→	None		Low		High		Noise→	None		Low		High		Noise→	None		Low		High	
Measure	**B**	**D**	**B**	**D**	**B**	**D**	**Measure**	**B**	**D**	**B**	**D**	**B**	**D**	**Measure**	**B**	**D**	**B**	**D**	**B**	**D**
MxA*	2	1	2	1	1	1	MxA[t]*	2	1	2	1	1	2	MxG*	2	1	2	1	1	2
MxG[t]*	2	1	2	1	1	1	MxL*	2	1	2	1	2	1	MxL[t]*	2	1	2	1	1	1
MxS*	2	1	2	1	2	1	MxS[t]*	2	1	2	1	2	1	SA*	2	1	2	1	1	1
SA[t]*	2	1	2	1	1	2	SG*	2	1	1	1	1	2	SG[t]*	2	1	2	1	1	1
SL*	2	1	2	1	2	1	SL[t]*	2	1	2	1	2	1	SS*	2	1	2	1	2	1
SS[t]*	2	1	2	1	2	1	S[t]A*	2	1	1	2	1	2	S[t]A[t2]*	2	1	1	2	1	2
S[t]G*	2	1	2	1	1	1	S[t]G[t2]*	2	1	2	1	1	1	S[t]L*	2	1	2	1	1	1
S[t]L[t2]*	2	1	2	1	1	1	S[t]S*	2	1	2	1	2	1	S[t]S[t2]*	2	1	2	1	2	1
SqA*	2	1	1	1	1	2	SqA[t]*	2	1	1	2	1	2	SqG*	2	1	1	1	1	2
SqG[t]*	2	1	2	1	1	1	SqL*	2	1	2	1	1	1	SqL[t]*	2	1	2	1	1	1
SqS*	2	1	2	1	1	1	SqS[t]*	2	1	2	1	1	1	Sq[t]A*	1	2	1	2	1	2
Sq[t]A[t2]*	1	2	1	2	1	2	Sq[t]G*	2	1	2	1	1	2	Sq[t]G[t2]*	2	1	2	1	1	2
Sq[t]L*	2	1	2	1	1	2	Sq[t]L[t2]*	2	1	2	1	1	2	Sq[t]S*	2	1	2	1	1	2
Sq[t]S[t2]*	2	1	1	1	1	2	SwA*	2	1	2	1	1	1	SwA[t]*	2	1	2	1	1	2
SwG*	2	1	2	1	1	1	SwG[t]*	2	1	2	1	1	1	SwL*	2	1	2	1	2	1
SwL[t]*	2	1	2	1	2	1	SwS*	2	1	2	1	2	1	SwS[t]*	2	1	2	1	2	1
Sw[t]A*	2	1	2	1	1	2	Sw[t]A[t2]*	2	1	2	1	1	2	Sw[t]G*	2	1	2	1	1	1
Sw[t]G[t2]*	2	1	2	1	1	1	Sw[t]L*	2	1	2	1	2	1	Sw[t]L[t2]*	2	1	2	1	2	1
Sw[t]S*	1	1	2	1	2	1	Sw[t]S[t2]*	2	1	2	1	2	1	SxA*	2	1	2	1	2	1
SxA[t]*	2	1	2	1	1	1	SxG*	2	1	2	1	2	1	SxG[t]*	2	1	2	1	2	1
SxL*	2	1	2	1	2	1	SxL[t]*	2	1	2	1	2	1	SxS*	2	1	2	1	2	1
SxS[t]*	2	1	2	1	2	1	Sx[t]A*	2	1	1	1	1	2	Sx[t]A[t2]*	2	1	1	1	1	2
Sx[t]G*	2	1	2	1	2	1	Sx[t]G[t2]*	2	1	2	1	2	1	Sx[t]L*	2	1	2	1	2	1
Sx[t]L[t2]*	2	1	2	1	2	1	Sx[t]S*	2	1	2	1	2	1	Sx[t]S[t2]*	2	1	2	1	2	1

Table 5.5: Over all ranking of channel functions for each match cost function at 95% confidence for each data set.

In this section we analyze their relative performance.

Table 5.4 presents our over all and per-framework rankings of the two channel functions on each data set. The data indicates that statistically significantly better results can almost always be expected from match cost functions using a difference function for a channel function on input with low magnitude or no noise. The data indicates that the choice of channel function makes less of a difference on input with high magnitude noise.

Table 5.5 and Tables C.1 through C.5 (see Appendix C) show the rankings of the channel functions for each combination of channel aggregate and channel norm function. With most match cost functions, the form that uses a difference channel function can be expected to yield statistically significantly better results than the form that uses a Birchfield and Tomasi channel function on input with low magnitude or no noise. When the magnitude of the noise in the input becomes high most match cost functions can be expected to yield statistically significantly better results with the a Birchfield and Tomasi channel function. Additionally, the Sq[t]A[t2]{B,D} and Sq[t]A{B,D} match cost functions can be expected to always produce statistically significantly better results in the version that uses the Birchfield and Tomasi measure as a channel function, and the S[t]A{B,D} and S[t]A[t]{B,D} match cost functions can also be expected to produce statistically significantly better results with a Birchfield and Tomasi measure on images containing noise.

5.3.3 Channel Norm Function Analysis

Our study of match cost functions uses eight channel norm functions: the L1 norm (A), L2 norm (S), generalized exponential (G), Lorentzian p-function (L), and their truncated versions. In this section we analyze their relative performance, and the parameter values found by our optimizer.

Relative Performance

Table 5.6 contains the over all and per-framework rankings of the eight channel norm functions used in our study for each data set. The data suggests that we can expect match cost functions that use a truncated generalized exponential to universally yield the best performing match cost functions. The data also indicates that the Lorentzian p-function can usually be expected to yield match cost functions with performance better than the L2 norm. This aggregate ranking data also suggests that the choice of channel norm function makes little statistical difference within some stereopsis frameworks in the presence of sensor noise.

Recall that the L1 and L2 norms are special cases of the generalized exponential. In this data the generalized exponential almost always outperforms the L1 norm which, in turn, typically outperforms the L2 norm. This suggests that the ideal choice of exponent in the generalized exponential is closer to one (as in the L1 norm) than two (as in the L2 norm); specifically, an exponent in the range $(0, 1) \cup (1, 1.5)$. We delve deeper into the ideal choice of parameter for the generalized exponential below.

Table 5.7 contains the over all rank of each channel norm function for each combination of channel aggregate and channel function. Tables D.1 through D.5 (see Appendix D) contain the rankings of the channel norm functions for each combination of channel aggregate and channel function for each global stereopsis framework. Looking closely at the ranking data in these tables, there appears to be some sort of correlation between the rankings of the L, L[t], S, and S[t] channel norm functions.

The Generalized Exponential Exponent

Figures 5.1, 5.2, and 5.3 present the distributions of the optimal generalized exponential exponent selected by the parameter optimizer for each match cost function that utilized

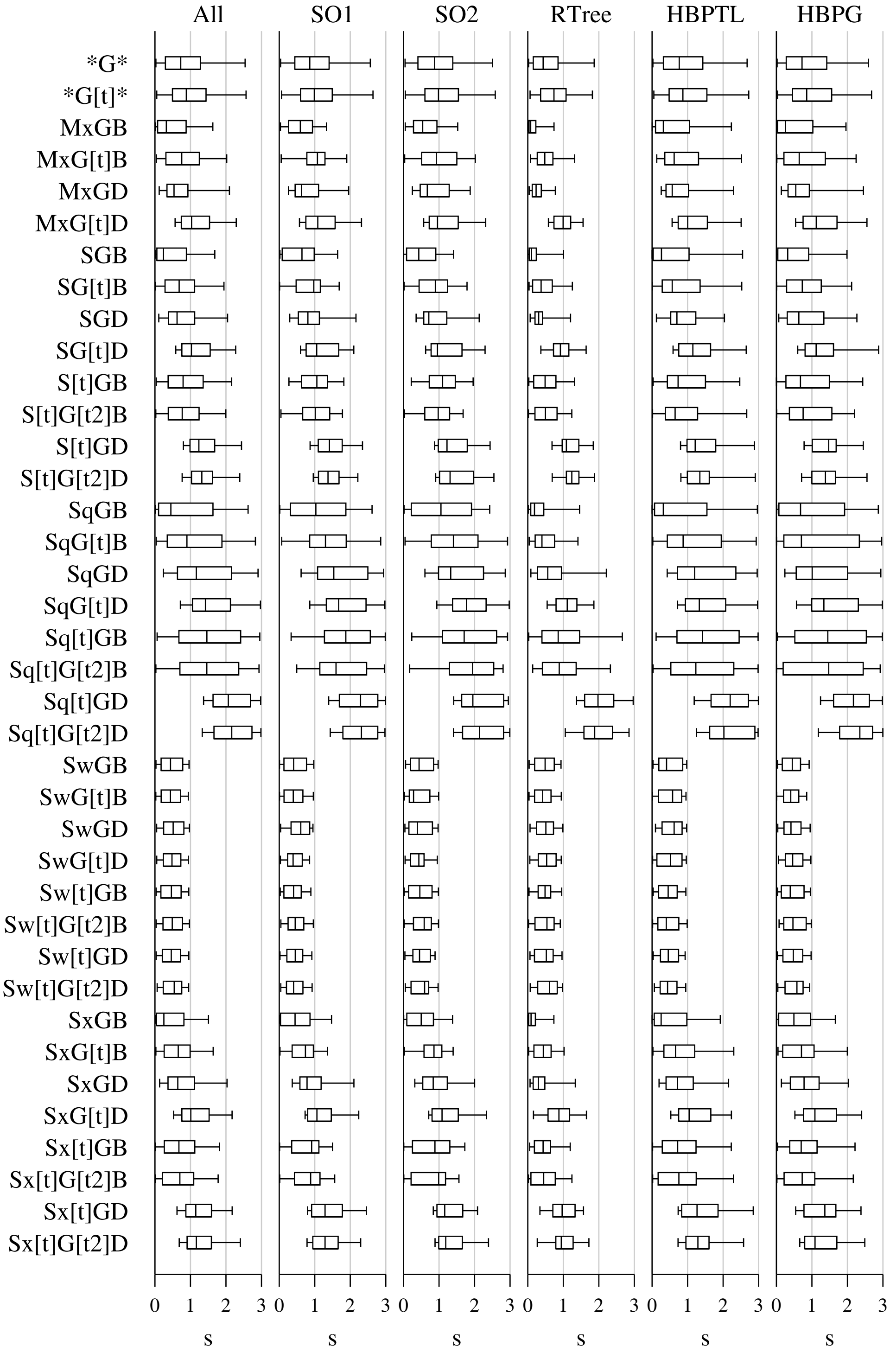


Figure 5.1: Box plots showing the distributions of optimal generalized exponential exponent on the data set with no noise introduced. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution. See Section 2.2.2 for an explanation of the notation used for match cost functions.

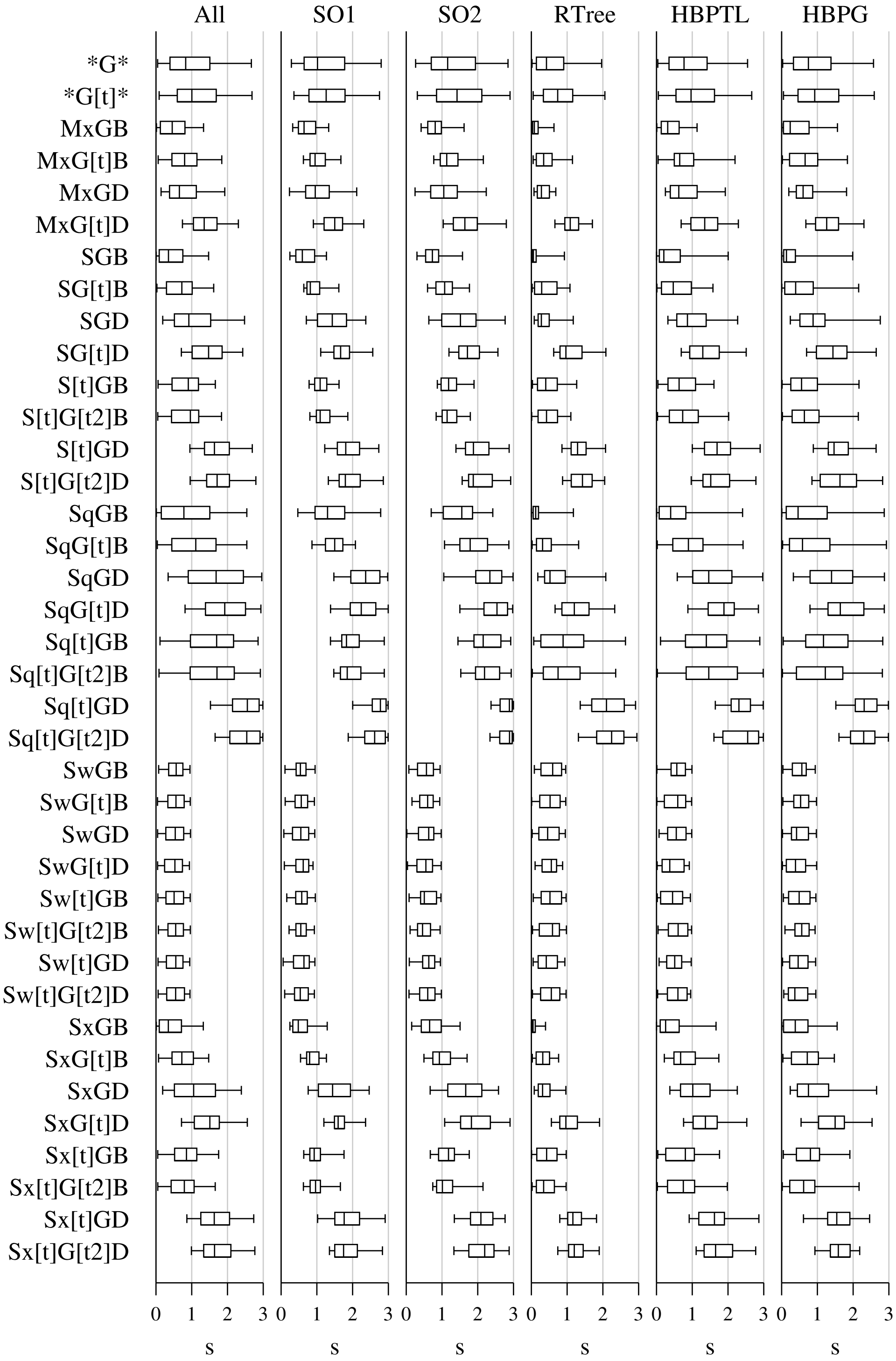


Figure 5.2: Box plots showing the distributions of optimal generalized exponential exponent on the data set with low magnitude noise. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution. See Section 2.2.2 for an explanation of the notation used for match cost functions.

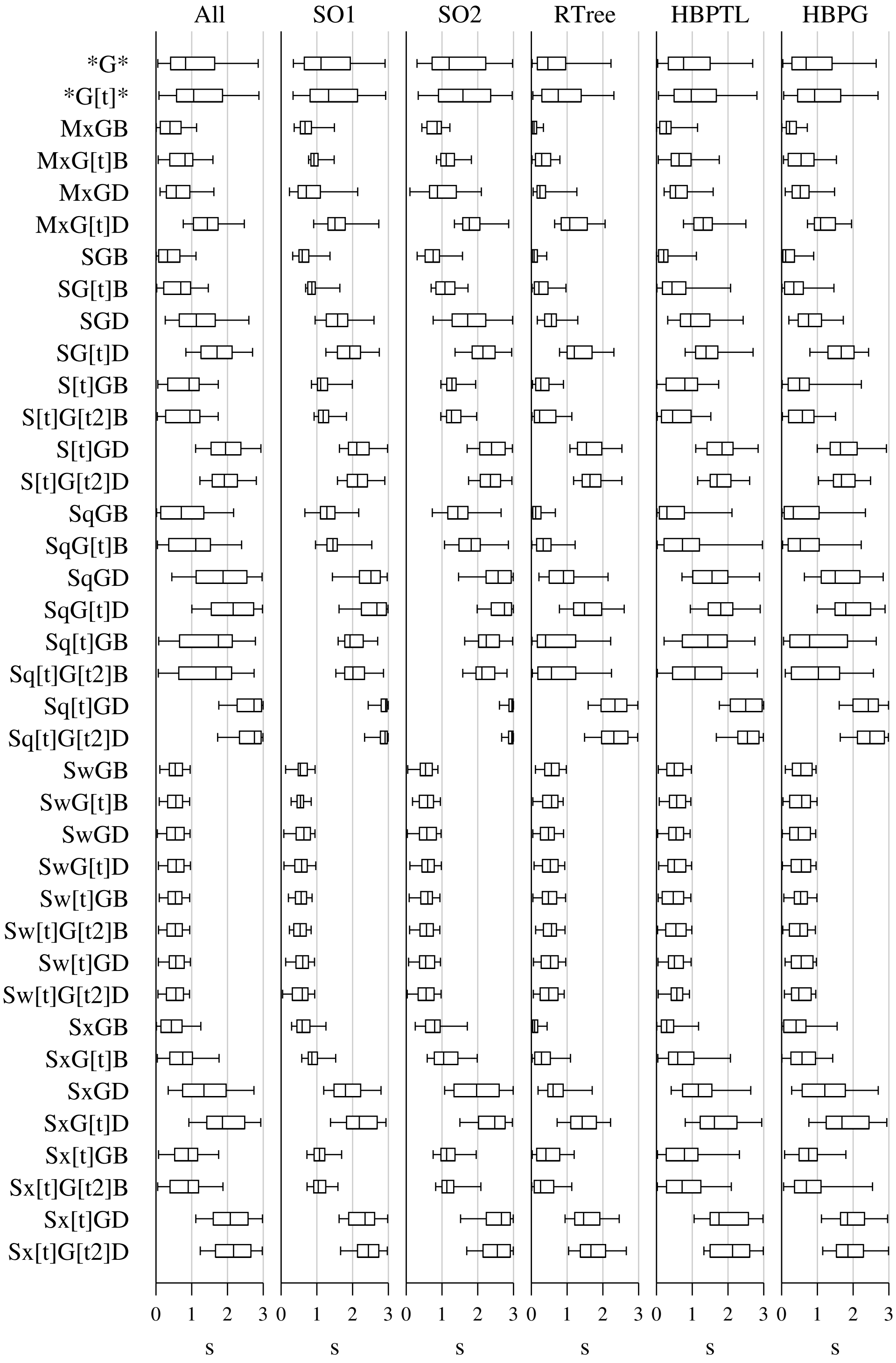


Figure 5.3: Box plots showing the distributions of optimal generalized exponential exponent on the data set with high magnitude noise. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution. See Section 2.2.2 for an explanation of the notation used for match cost functions.

Channel norm function	Over all	SO1	SO2	RTree	HBPTL	HBPG
A	4	3	2	1	3	3
A[t]	3	3	2	1	2	2
G	2	2	2	1	2	2
G[t]	1	1	1	1	1	1
L	5	3	3	2	4	3
L[t]	4	3	3	1	3	2
S	7	5	5	3	6	5
S[t]	6	4	4	2	5	4

(a) Rankings for data set with no noise introduced.

Channel norm function	Over all	SO1	SO2	RTree	HBPTL	HBPG
A	2	2	2	1	2	2
A[t]	2	2	2	1	2	1
G	1	1	1	1	1	1
G[t]	1	1	1	1	1	1
L	2	1	1	2	3	2
L[t]	1	1	1	1	2	1
S	3	2	1	2	4	3
S[t]	2	1	1	1	3	3

(b) Rankings for data set with lowest magnitude noise introduced.

Channel norm function	Over all	SO1	SO2	RTree	HBPTL	HBPG
A	4	3	2	1	2	1
A[t]	5	3	3	1	1	1
G	1	1	1	1	1	1
G[t]	1	1	1	1	1	1
L	2	2	1	1	2	2
L[t]	2	2	1	1	1	1
S	4	2	1	2	3	3
S[t]	3	2	1	1	3	2

(c) Rankings for data set with highest magnitude noise introduced.

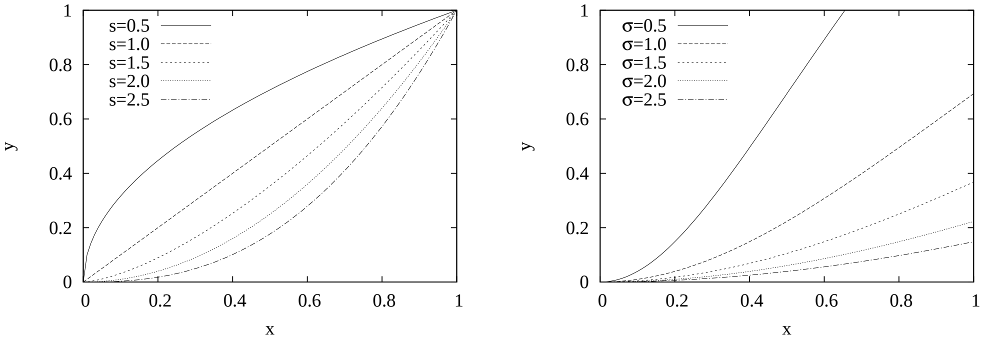
Table 5.6: Cluster rankings of channel norm functions at 95% confidence. Rankings displayed are the over all rankings, and the overall rankings for each global stereopsis framework.

either the generalized exponential (G) or truncated generalized exponential (G[t]) channel norm function. The distributions are separated into columns representing the distribution restricted to each stereopsis framework or the full distribution (in the "All" column). The left and right ends of the box tails indicate the fifth and 95^{th} percentiles, the box extents indicate the 25^{th} and 75^{th} percentiles, and the vertical line within the box indicates the median of the distribution. Furthermore, the "*G*" and "*G[t]*" rows are the aggregate distributions over all match cost functions XGZ and XG[t]Z for channel aggregate X and channel function Z, respectively. These figures can be useful to researchers and practitioners to narrow down the range of parameter values to consider when using a generalized exponential channel norm function.

As the magnitude of noise in the data set is increased, the optimal choice of exponent for the G and G[t] channel norm functions exhibits a subtle, but distinctive, shift to higher values. Consider the graphs of the generalized exponential in Figure 5.4a. Notice that as the value of the exponent increases in value that the function becomes more flat around $x = 0$. This suggests that as the magnitude of the noise in a data set is increased that channel norm

Noise→	**None**								**Low Magnitude**								**High Magnitude**							
Measure	**A**	**A[t]**	**G**	**G[t]**	**L**	**L[t]**	**S**	**S[t]**	**A**	**A[t]**	**G**	**G[t]**	**L**	**L[t]**	**S**	**S[t]**	**A**	**A[t]**	**G**	**G[t]**	**L**	**L[t]**	**S**	**S[t]**
Mx*B	3	1	2	1	5	2	6	4	3	1	2	2	5	4	6	5	2	1	1	2	4	3	5	4
Mx*D	3	1	2	1	3	2	5	4	1	1	2	1	3	2	4	2	1	4	1	2	3	3	4	2
S*B	3	1	2	1	5	2	6	4	3	1	3	2	5	4	6	5	2	1	1	1	4	3	5	4
S*D	3	1	2	1	3	2	5	4	2	1	2	1	2	1	2	1	2	3	2	1	1	1	1	1
S[t]*B	1	1	1	1	1	2	3	3	1	1	2	2	3	3	4	4	1	2	3	2	4	4	5	5
S[t]*D	1	1	1	1	2	2	3	3	2	2	1	1	1	1	1	1	3	4	2	2	2	2	1	1
Sq*B	2	2	2	1	3	2	3	3	1	1	1	1	2	2	2	2	1	1	1	1	3	2	2	2
Sq*D	2	1	2	1	3	1	4	2	3	3	2	1	2	1	2	1	3	4	2	1	2	2	1	1
Sq[t]*B	2	2	1	1	1	1	1	1	3	3	1	1	1	2	1	1	3	3	1	2	1	2	1	1
Sq[t]*D	3	3	1	1	1	2	1	1	3	3	1	1	1	1	2	2	3	4	1	1	1	1	2	2
Sw*B	2	1	2	1	4	2	5	3	2	1	2	2	5	3	6	4	2	1	2	2	5	3	5	4
Sw*D	4	1	3	2	5	4	7	6	2	2	3	2	2	2	2	1	3	3	3	1	2	2	1	1
Sw[t]*B	1	1	1	1	2	2	3	3	1	2	2	3	4	4	5	5	1	1	1	2	3	3	3	4
Sw[t]*D	1	1	1	1	2	2	3	3	1	1	1	2	1	2	1	1	3	4	1	2	1	2	1	1
Sx*B	3	1	2	1	5	2	6	4	3	1	3	2	6	4	7	5	2	1	2	2	4	3	5	4
Sx*D	4	1	3	2	4	4	6	5	3	2	3	1	2	1	1	1	3	4	2	1	1	1	1	1
Sx[t]*B	1	1	1	1	2	2	3	4	1	1	2	2	3	3	3	3	1	1	2	2	3	3	4	4
Sx[t]*D	1	1	1	1	2	3	4	4	4	5	2	3	2	2	1	1	3	4	1	1	1	2	1	1

Table 5.7: Over all ranking of channel norm functions for each match cost function at 95% confidence for each data set.



(a) Generalized exponential.

(b) Lorentzian p-function.

Figure 5.4: Plots of the generalized exponential and Lorentzian p-function with varying parameter values. See Section 2.2.2 for an explanation of the notation used for match cost functions.

functions that are less discriminating around $x = 0$ are preferred. Furthermore, the range of optimal parameters tends to shrink in size for many match cost functions as the magnitude of noise in the data set is increased. This range shrinking is particularly pronounced in the SO1 and SO2 framework, and with many of the match cost functions that incorporate the Birchfield and Tomasi channel function.

In general, the optimal choice of exponent for the G and G[t] channel norm functions is clustered near a value of one. However, the range of the optimal exponent can differ quite drastically between match cost functions. For example, the ranges of optimal exponents for the SqG* and Sq[t]G* match cost functions do not overlap at all with the ranges from any of the match cost functions that incorporate a weighted sum (Sw) or truncated weighted sum (Sw[t]) channel aggregate. Without a doubt, the flexibility imparted by the parameterization of the exponent in the generalized exponential is what allows it to outperform both the L1 and L2 norm (both special cases of the generalized exponential) channel norm functions.

Lorentzian p-function User-Parameter

Figures 5.5, 5.6, and 5.7 present the distributions of the optimal scaling parameter in the Lorentzian p-function selected by the parameter optimizer. Distributions are presented for each match cost function that utilized either the Lorentzian p-function (L) or truncated Lorentzian p-function (L[t]) channel norm function. We also present aggregate distributions *L* and *L[t]* for all match cost functions XLZ and XL[t]Z, respectively. These distributions display data in the same manner as the box plots for the generalized exponential exponent, described above.

Interestingly, the scaling parameter in the Lorentzian p-function channel norm function exhibits many of the same tendencies as the exponent parameter in the generalized exponential channel norm function. As the magnitude of the noise in the data set is increased, the optimal range for this scaling parameter exhibits a subtle, but distinctive, shift to higher values. As with the generalized exponential exponent, the range of optimal parameters shrinks as the magnitude of noise in the data set is increased. Additionally, the optimal ranges for the scaling parameter in the SqL* and Sq[t]L* match cost functions do not overlap with the ranges from any of the match cost functions that incorporate a Sw or Sw[t] channel aggregate.

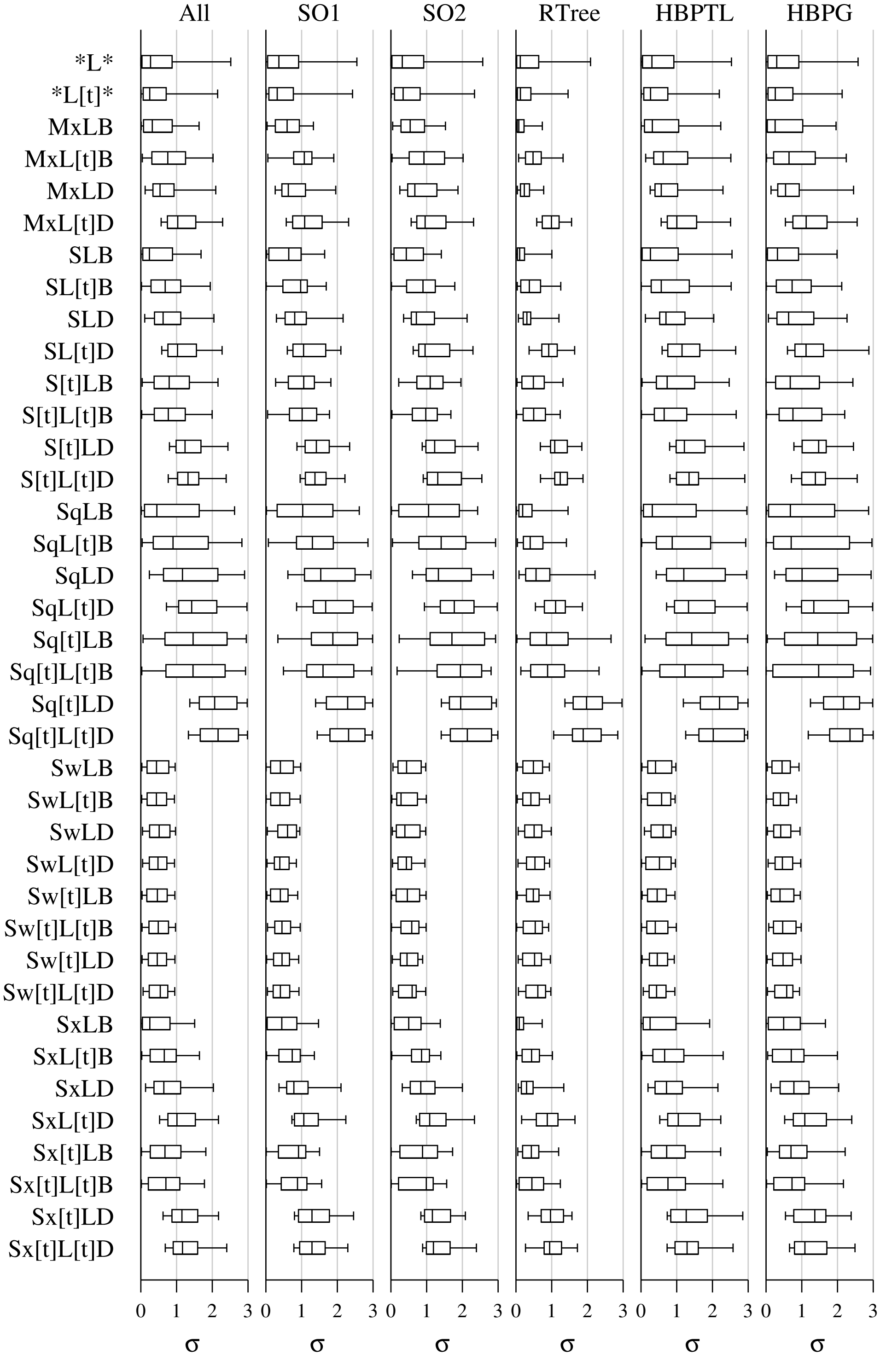


Figure 5.5: Box plots showing the distributions of the optimal Lorentzian p-function channel norm function's parameter on the data set with no noise introduced. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution. See Section 2.2.2 for an explanation of the notation used for match cost functions.

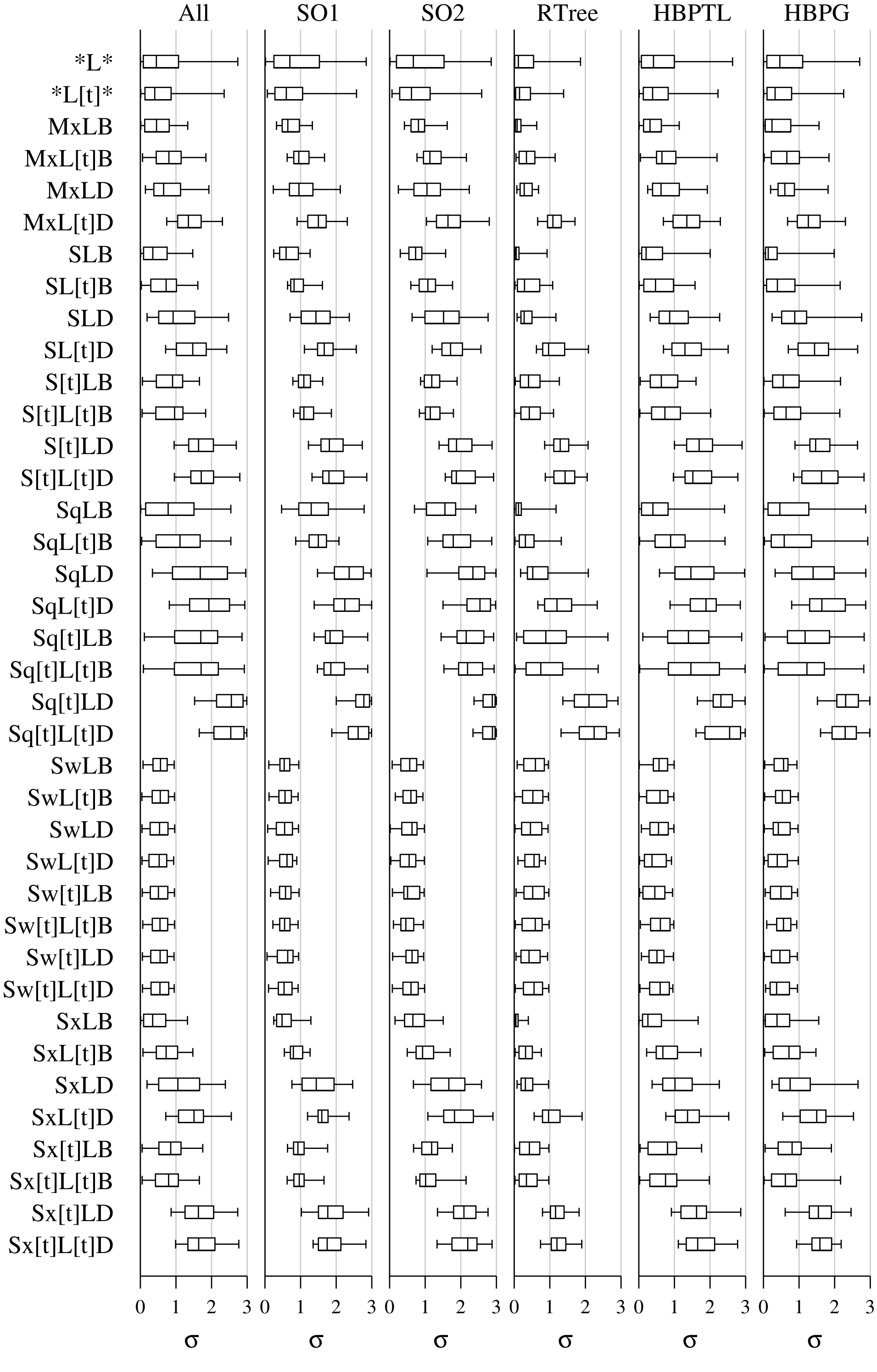


Figure 5.6: Box plots showing the distributions of the optimal Lorentzian p-function channel norm function's parameter on the data set with low magnitude noise. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution. See Section 2.2.2 for an explanation of the notation used for match cost functions.

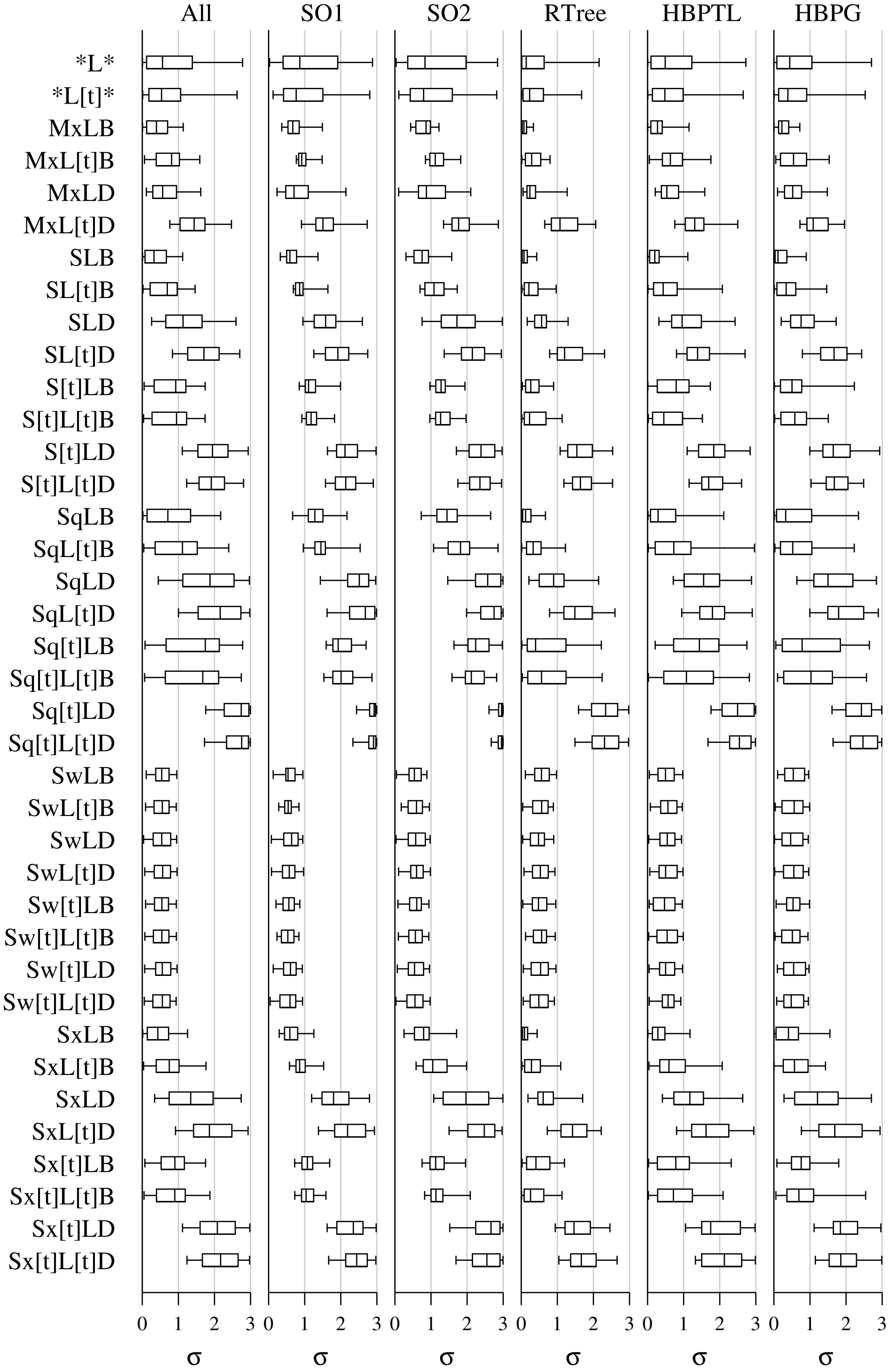


Figure 5.7: Box plots showing the distributions of the optimal Lorentzian p-function channel norm function's parameter on the data set with high magnitude noise. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution. See Section 2.2.2 for an explanation of the notation used for match cost functions.

Correlation Between Generalized Exponential and Lorentzian p-function

Consider the graphs of the generalized exponential, $g(x) = |x/\sigma|^s$, with $\sigma = 1$ and the Lorentzian p-function, $l(x) = \ln(1 + (\frac{x}{\sigma})^2)$, in Figures 5.4a and 5.4b, respectively. Notice that as the exponent, s, in the generalized exponential approaches infinity that the curve of $g(x)$ becomes more flat near $x = 0$. Furthermore, notice that as the scaling factor, σ, in the Lorentzian p-function decreases that the shape of $l(x)$ begins to vaguely resemble the shape of $g(x)$ for low values of s.

Scrutinizing the ranking data in Table 5.7, and the optimal parameter distributions in Figures 5.1, 5.2, 5.3, 5.5, 5.6, and 5.7 we see that there is a relationship between the ranking of the Lorentzian p-function and generalized exponential channel norm functions with the distributions of their optimal parameter values. Specifically, when the distributions of optimal s and σ are such that the resulting $g(x)$ and $l(x)$ will be similar then the rank of the Lorentzian is similar to the rank of the generalized exponential; this occurs when s and σ have low values. On the other hand, when the distributions of optimal s and σ are such that the two curves will not be similar then the Lorentzian p-function ranks poorly compared to the generalized exponential. Furthermore, this relationship appears to be governed by the choice of channel aggregate used in the match cost function.

5.3.4 Channel Aggregate Analysis

In this study we use nine channel aggregates: max (Mx), sum (S), square root of sum (Sq), weighted sum, sum minus max (Sx), and the truncated forms of all except the max aggregate. In this section we analyze their relative performance, and the parameter values found by our optimizer for the weighted sum aggregate.

Relative Performance

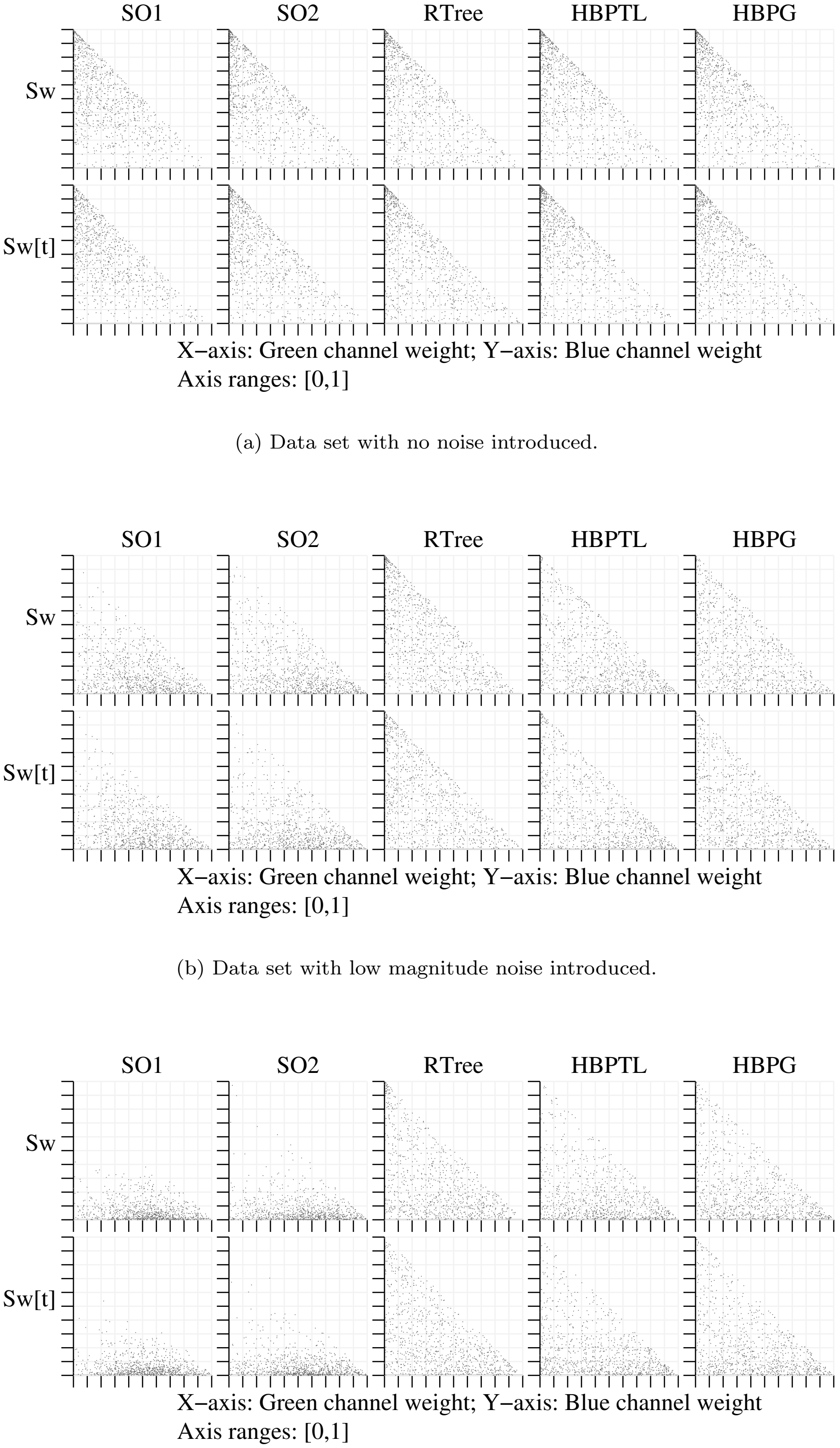
Table 5.8 contains the over all and per-framework rankings of the nine channel aggregates used in our study for each data set. This data indicates that we can generally expect good performance from match cost functions based on the max (Mx), truncated sum (S[t]), and square root of sum (Sq) channel aggregates on input with no noise. On input with low magnitude noise we can expect match cost functions based on the Mx, Sq, and truncated weighted sum (Sw[t]) channel aggregates to perform well. However, on input with high magnitude sensor noise only match cost functions based on the the Sw and Sw[t] channel aggregates can be generally expected to perform the best.

The parameters in the weighted sum channel aggregate allow the match cost function to assign more weight to the colour channels that are more important for match cost calculation. Thus, if the relative importance of the colour channels changes, as seems to be the case as the magnitude of sensor noise increases, this channel aggregate's flexibility can be a great help.

Additional data on the rankings of each channel aggregate for each combination of channel norm function and channel norm can be found in table 5.9 and Tables E.1 through E.5. Table 5.9 contains the over all rankings, and Tables E.1 through E.5 (see Appendix E) contain the per-framework rankings.

User-parameters

Figure 5.8 contains a series of two dimensional scatter plots of the optimized green and blue colour channel weights for the weighted sum channel aggregates. The scatter plots are divided into two groups: those for the weighted sum channel aggregate (Sw), and those for its truncated form (Sw[t]). The scatter plots contain all 912 data points optimized for the 16 match cost functions that use the channel aggregate indicated in each stereopsis framework.



(a) Data set with no noise introduced.

(b) Data set with low magnitude noise introduced.

(c) Data set with high magnitude noise introduced.

Figure 5.8: Scatter plots of the blue and green weights for the weighted sum channel aggregate for each framework.

Channel aggregate	Over all	SO1	SO2	RTree	HBPTL	HBPG
Mx	1	1	1	2	1	2
S	3	2	3	2	2	3
S[t]	1	2	2	1	1	1
Sq	1	2	1	1	1	2
Sq[t]	2	3	3	1	1	1
Sw	4	3	3	2	2	3
Sw[t]	2	2	2	1	1	2
Sx	6	5	5	2	4	5
Sx[t]	5	4	4	1	3	4

(a) Rankings for data set with no noise introduced.

Channel aggregate	Over all	SO1	SO2	RTree	HBPTL	HBPG
Mx	1	1	1	1	1	1
S	2	2	1	1	2	2
S[t]	2	3	2	1	1	1
Sq	1	1	1	1	1	1
Sq[t]	3	3	3	1	1	1
Sw	2	1	1	1	2	3
Sw[t]	1	1	1	1	1	1
Sx	5	4	3	1	4	5
Sx[t]	4	4	3	1	3	4

(b) Rankings for data set with lowest magnitude noise introduced.

Channel aggregate	Over all	SO1	SO2	RTree	HBPTL	HBPG
Mx	3	3	2	1	1	1
S	4	4	2	1	2	2
S[t]	4	5	3	1	1	1
Sq	2	2	2	1	1	1
Sq[t]	5	4	4	1	1	1
Sw	1	1	1	1	1	1
Sw[t]	1	1	1	1	1	1
Sx	6	6	4	2	3	3
Sx[t]	7	7	4	1	3	3

(c) Rankings for data set with highest magnitude noise introduced.

Table 5.8: Cluster rankings of channel aggregates at 95% confidence. Rankings displayed are the over all rankings, and the overall rankings for each global stereopsis framework.

When the input contains no noise there is a tendency for the optimized weight for the blue colour channel to be closer to one. Indicating that the most useful information about a match's quality is contained in the blue colour channel for this data set.

However, in the SO1 and SO2 stereopsis frameworks there is a distinct trend to lower blue-channel weights as the magnitude of the noise in the data set is increased. This trend is less drastic within the other three frameworks, but is still present.

Consulting the noise variance curves in Figure 4.2 we see that the green channel tends to have the lowest variance noise, and the blue channel tends to have the highest variance noise. The difference between the two variance curves tends to be less than 0.02, or around five eight-bit grey levels; this difference would be about 0.04, or around 10 eight-bit grey levels in the high magnitude noise data set. Thus, the evidence suggests that even a small difference in noise variance between colour channels can cause a change in the relative importance of those colour channels for high quality match cost functions.

Measure	**Mx**	**S**	**S[t]**	**Sq**	**Sq[t]**	**Sw**	**Sw[t]**	**Sx**	**Sx[t]**
*AB	2	3	1	2	3	2	1	4	3
*AD	2	3	1	2	5	3	1	4	2
*A[t]B	1	2	2	2	3	2	2	4	2
*A[t]D	1	2	2	2	4	2	2	3	3
*GB	2	2	1	2	1	3	1	4	3
*GD	2	4	2	3	1	4	3	5	2
*G[t]B	1	2	2	2	2	2	3	4	3
*G[t]D	1	2	2	2	1	3	3	4	3
*LB	3	4	1	2	1	4	2	5	2
*LD	2	3	2	2	1	4	3	5	2
*L[t]B	1	2	1	1	1	2	2	3	3
*L[t]D	1	2	3	1	1	4	3	5	3
*SB	4	4	3	2	1	4	3	5	3
*SD	4	5	2	2	1	5	3	6	4
*S[t]B	3	4	3	2	1	4	3	5	3
*S[t]D	4	5	3	2	1	5	5	6	5

(a) Rankings for data set with no noise introduced.

Measure	**Mx**	**S**	**S[t]**	**Sq**	**Sq[t]**	**Sw**	**Sw[t]**	**Sx**	**Sx[t]**
*AB	2	3	1	1	4	2	1	4	2
*AD	1	2	3	2	4	2	1	3	3
*A[t]B	1	1	1	1	3	1	2	3	2
*A[t]D	1	2	5	3	6	2	1	4	3
*GB	1	2	2	1	1	2	1	3	2
*GD	2	3	2	2	1	3	2	4	3
*G[t]B	1	2	2	1	1	2	2	3	2
*G[t]D	1	2	1	1	1	2	2	3	3
*LB	4	4	2	2	1	4	3	5	3
*LD	2	3	2	1	1	2	1	4	4
*L[t]B	2	2	2	1	1	2	2	3	2
*L[t]D	2	2	1	1	1	1	2	3	3
*SB	4	5	3	2	1	4	3	6	4
*SD	2	2	1	1	1	1	1	3	4
*S[t]B	3	3	3	2	1	3	3	4	2
*S[t]D	1	1	1	1	1	1	1	2	4

(b) Rankings for data set with lowest magnitude noise introduced.

Measure	**Mx**	**S**	**S[t]**	**Sq**	**Sq[t]**	**Sw**	**Sw[t]**	**Sx**	**Sx[t]**
*AB	3	4	3	1	5	2	1	5	4
*AD	1	2	4	2	5	1	1	3	3
*A[t]B	2	3	4	2	5	1	1	5	2
*A[t]D	3	3	6	4	7	1	2	5	4
*GB	1	3	3	1	2	1	1	4	2
*GD	2	4	3	2	3	2	1	5	3
*G[t]B	2	3	3	1	1	1	1	4	2
*G[t]D	3	3	3	2	3	1	2	4	3
*LB	4	5	3	2	1	3	2	6	3
*LD	3	3	3	2	2	1	1	4	2
*L[t]B	3	4	4	1	1	2	2	5	3
*L[t]D	3	2	3	2	3	1	1	4	2
*SB	6	6	5	2	1	4	3	7	4
*SD	4	3	2	2	5	1	1	4	2
*S[t]B	4	5	4	2	1	3	3	6	4
*S[t]D	4	4	4	3	6	1	2	5	3

(c) Rankings for data set with highest magnitude noise introduced.

Table 5.9: Over all ranking of channel aggregates for each match cost function at 95% confidence.

For the interested reader, Figures F.1 through F.12 in Appendix F contain two dimensional scatter plots of the optimized green and blue weights for each match cost function constructed from the weighted sum and truncated weighted sum channel aggregates.

5.4 Spatially Aggregated Match Cost Functions

In the second phase of our study on match cost functions for global stereopsis algorithms we compare the relative performance of the five spatial aggregates listed in Table 2.1d. Each of these spatial aggregates are applied to the four match cost functions S[t]AD, MxA[t]D, MxG[t]D, and Sq[t]GD. Furthermore, all of our spatial aggregates sum the values of a given match cost function over a square $(2r + 1) \times (2r + 1)$ window; as is the norm in practice.

The value used for the radius, r, can have a drastic effect on the results produced by a given spatial aggregate. For some spatial aggregates a small window is preferable, and for others a large window is preferred. Since we do not know what the ideal choice of window size is for each of our spatial aggregates we use eight different values for r in each of our spatial aggregates: 1, 3, 5, 7, 9, 11, 13, and 15. This selection of values for r covers the vast majority of window sizes used in the literature. The result is a set of 160 match cost functions to compare in this phase of our study.

The definitions of the *spatial weight* (Ws), *color weight* (Wc), *bilateral* (B), and *adaptive support weight* (Wa) spatial aggregates all include user-defined parameters α and/or β. The α parameter is the scale factor applied to the argument of the Gaussian weight applied in the spatial dimension, and the β parameter is the scale factor applied to the argument of the Gaussian applied in the spectral (colour) domain. Throughout this study we use a value of α equal to the radius, r, of the spatial aggregation window, and we let our parameter optimizer find an optimal choice for β from the range $[0.1, 10]$. Furthermore, as suggested by Yoon and Kweon [105], the colour difference in the weight functions of the Wc, B, and Wa spatial aggregates is taken in the CIELuv colour space rather than the RGB colour space. Since the CIELuv space is perceptually uniform, the difference in this colour space will more accurately reflect the perceived difference between colours.

5.4.1 Comparison of Spatial Aggregates

Tables 5.10 and 5.11 present our over all and per-framework rankings of the 40 spatial aggregates (five spatial aggregates with eight spatial radii each) used in this study. We calculate these rankings using cluster ranking at 95% confidence with the same methodology we employ to calculate the rankings for the match cost function component rankings in the first phase of our study on match cost functions.

The ranking data from these two tables and Tables G.5 and G.6 indicates that the choice of spatial aggregate can not be expected to make any difference on the accuracy of algorithms that use the RTree framework. This is not entirely unexpected in light of the rankings seen in our analysis of the first phase of this study. Since the RTree formulation defines its energy function over a segmentation of the image, the accuracy of algorithms based on it will be more affected by the quality of the segmentation used than the match cost function.

The ranking data also indicates that the bilateral (B) spatial aggregate can be expected to yield the best results of all of the spatial aggregates we consider, provided that the spatial radius is chosen well, regardless of the amount of noise in the data set. A spatial radius of at least nine appears to be ideal, with larger radii required as the magnitude of the noise in the data set is increased.

To help gauge how much of an improvement can be expected as spatial radius is increased we include the mean accuracy improvements of each spatial aggregate over their accuracy with a spatial radius of one in Figures 5.10 and 5.11. These mean values are calculated from

Noise→	None						Low Magnitude						High Magnitude					
Spatial Aggregate	**All**	**SO1**	**SO2**	**RTree**	**HBPTL**	**HBPG**	**All**	**SO1**	**SO2**	**RTree**	**HBPTL**	**HBPG**	**All**	**SO1**	**SO2**	**RTree**	**HBPTL**	**HBPG**
M ($r = 1$)	12	6	7	1	8	5	12	6	7	1	8	5	13	9	8	1	6	4
M ($r = 3$)	14	7	8	1	9	6	9	5	5	1	8	5	8	6	6	1	5	3
M ($r = 5$)	16	9	10	1	11	7	11	6	7	1	10	6	8	5	6	1	5	4
M ($r = 7$)	19	10	11	1	12	9	15	7	8	1	11	7	9	6	6	1	6	4
M ($r = 9$)	21	12	13	1	14	11	17	8	10	1	13	8	11	6	7	1	7	5
M ($r = 11$)	23	14	15	1	16	13	19	10	12	1	15	9	13	7	8	1	8	6
M ($r = 13$)	25	16	17	1	18	15	21	11	14	1	16	11	14	8	8	1	9	7
M ($r = 15$)	26	17	18	1	19	16	22	12	15	1	17	12	15	9	9	1	10	8
Ws ($r = 1$)	11	6	5	1	7	4	13	7	7	1	8	6	14	11	10	1	6	5
Ws ($r = 3$)	13	6	7	1	8	5	8	5	5	1	7	5	9	6	7	1	4	3
Ws ($r = 5$)	15	8	9	1	9	6	9	5	6	1	8	5	8	5	6	1	6	3
Ws ($r = 7$)	16	9	10	1	10	7	10	6	7	1	9	6	8	5	6	1	6	4
Ws ($r = 9$)	18	10	11	1	12	8	14	7	8	1	10	7	9	5	6	1	6	4
Ws ($r = 11$)	20	11	12	1	13	10	16	8	9	1	12	8	10	6	7	1	7	5
Ws ($r = 13$)	22	13	14	1	15	12	18	9	11	1	14	9	12	6	7	1	7	6
Ws ($r = 15$)	24	15	16	1	17	14	20	10	13	1	15	10	13	7	8	1	8	6
Wc ($r = 1$)	10	5	5	1	6	4	12	6	7	1	8	5	14	10	9	1	7	5
Wc ($r = 3$)	5	2	2	1	2	3	6	4	4	1	4	3	8	6	6	1	4	3
Wc ($r = 5$)	3	1	1	1	2	2	4	2	2	1	3	2	6	4	4	1	3	2
Wc ($r = 7$)	1	1	1	1	1	2	3	1	2	1	2	1	4	2	3	1	2	1

Table 5.10: Cluster rankings of spatial aggregates at 95% confidence for each data set, Table 1 of 2. Rankings displayed are the over all and per-framework rankings.

Noise→	None						Low Magnitude						High Magnitude					
Spatial Aggregate	**All**	**SO1**	**SO2**	**RTree**	**HBPTL**	**HBPG**	**All**	**SO1**	**SO2**	**RTree**	**HBPTL**	**HBPG**	**All**	**SO1**	**SO2**	**RTree**	**HBPTL**	**HBPG**
Wc ($r = 9$)	2	1	1	1	1	1	2	1	1	1	1	1	3	2	2	1	1	1
Wc ($r = 11$)	4	2	2	1	2	2	2	1	1	1	1	1	2	1	2	1	1	1
Wc ($r = 13$)	5	2	2	1	2	2	3	1	1	1	1	1	2	2	1	1	1	1
Wc ($r = 15$)	6	3	3	1	3	3	4	2	2	1	3	2	3	2	2	1	2	2
B ($r = 1$)	12	6	7	1	7	5	14	8	8	1	8	6	15	12	11	1	8	6
B ($r = 3$)	6	3	3	1	3	3	7	4	4	1	5	3	9	6	7	1	4	3
B ($r = 5$)	4	2	2	1	2	2	5	2	3	1	3	2	7	4	5	1	3	2
B ($r = 7$)	2	1	1	1	1	2	3	2	2	1	2	1	5	2	3	1	2	2
B ($r = 9$)	1	1	1	1	1	1	2	1	1	1	1	1	3	2	2	1	2	1
B ($r = 11$)	1	1	1	1	1	1	1	1	1	1	1	1	2	1	2	1	1	1
B ($r = 13$)	2	2	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
B ($r = 15$)	4	2	2	1	2	2	2	1	1	1	1	1	1	1	1	1	1	1
Wa ($r = 1$)	17	8	8	1	8	6	16	10	10	1	10	7	16	13	12	1	9	6
Wa ($r = 3$)	9	5	4	1	5	4	8	5	5	1	6	5	13	9	8	1	6	5
Wa ($r = 5$)	8	4	3	1	4	3	6	3	3	1	4	4	8	6	6	1	4	3
Wa ($r = 7$)	7	3	3	1	3	3	5	2	2	1	3	2	7	4	5	1	3	3
Wa ($r = 9$)	6	3	3	1	3	3	4	1	2	1	3	2	5	3	3	1	2	2
Wa ($r = 11$)	6	3	3	1	3	3	3	1	2	1	2	2	3	2	2	1	2	2
Wa ($r = 13$)	7	3	3	1	3	3	3	1	1	1	2	2	3	2	2	1	2	2
Wa ($r = 15$)	7	3	6	1	3	3	3	1	1	1	2	2	2	2	2	1	1	1

Table 5.11: Cluster rankings of spatial aggregates at 95% confidence for each data set, Table 2 of 2. Rankings displayed are the over all and per-framework rankings.

Noise→	**None**				**Low Magnitude**				**High Magnitude**			
Spatial Aggregate	**S[t]AD**	**MxA[t]D**	**MxG[t]D**	**Sq[t]GD**	**S[t]AD**	**MxA[t]D**	**MxG[t]D**	**Sq[t]GD**	**S[t]AD**	**MxA[t]D**	**MxG[t]D**	**Sq[t]GD**
M ($r = 1$)	10	11	10	10	11	11	12	10	14	13	12	11
M ($r = 3$)	11	12	11	11	10	10	11	8	11	9	9	8
M ($r = 5$)	13	14	13	13	12	11	12	12	10	8	9	9
M ($r = 7$)	14	16	15	15	14	14	15	14	10	10	9	10
M ($r = 9$)	16	18	17	17	16	16	17	16	12	12	11	11
M ($r = 11$)	18	20	19	20	18	18	19	18	13	14	12	12
M ($r = 13$)	20	22	21	21	20	20	21	20	14	15	13	13
M ($r = 15$)	21	23	22	22	21	21	22	21	15	16	14	14
Ws ($r = 1$)	10	10	9	9	12	12	13	11	16	15	13	12
Ws ($r = 3$)	9	11	10	10	7	9	9	7	11	9	9	8
Ws ($r = 5$)	12	13	12	12	8	9	10	8	8	8	8	7
Ws ($r = 7$)	13	14	13	13	12	11	12	9	8	8	8	8
Ws ($r = 9$)	14	15	14	14	13	13	14	13	10	9	9	9
Ws ($r = 11$)	15	17	16	16	15	15	16	15	11	11	10	10
Ws ($r = 13$)	17	19	18	18	17	17	18	17	12	13	11	11
Ws ($r = 15$)	19	21	20	19	19	19	20	19	13	14	12	12
Wc ($r = 1$)	8	9	8	8	11	11	12	10	15	14	13	12
Wc ($r = 3$)	4	4	4	4	5	6	7	5	10	8	8	6
Wc ($r = 5$)	2	3	3	3	4	4	5	4	6	7	6	5
Wc ($r = 7$)	1	2	2	2	3	2	3	3	4	5	4	3

Table 5.12: Over all rankings of each spatial aggregate for each base match cost function at 95% confidence for each data set, Table 1 of 2. Rankings are oriented vertically.

Noise→	None				Low Magnitude				High Magnitude			
Spatial Aggregate	**S[t]AD**	**MxA[t]D**	**MxG[t]D**	**Sq[t]GD**	**S[t]AD**	**MxA[t]D**	**MxG[t]D**	**Sq[t]GD**	**S[t]AD**	**MxA[t]D**	**MxG[t]D**	**Sq[t]GD**
Wc ($r = 9$)	2	2	2	2	2	1	2	2	2	3	3	2
Wc ($r = 11$)	3	3	3	3	2	2	2	2	2	3	2	2
Wc ($r = 13$)	3	4	4	4	2	2	3	2	2	2	1	2
Wc ($r = 15$)	4	5	5	5	4	4	4	4	2	3	2	2
B ($r = 1$)	10	10	9	9	13	13	14	12	17	16	14	13
B ($r = 3$)	4	5	5	5	6	7	8	6	11	8	9	7
B ($r = 5$)	3	3	3	3	4	5	6	4	7	7	7	5
B ($r = 7$)	1	2	1	2	3	3	3	3	4	6	5	3
B ($r = 9$)	1	1	1	1	2	2	2	1	3	4	3	2
B ($r = 11$)	1	1	1	1	1	1	1	1	2	2	2	1
B ($r = 13$)	1	2	2	2	1	1	1	1	1	2	1	1
B ($r = 15$)	2	3	3	2	1	1	1	1	1	1	1	1
Wa ($r = 1$)	11	12	11	11	18	15	17	15	18	17	15	15
Wa ($r = 3$)	7	8	8	7	9	8	10	7	14	13	12	11
Wa ($r = 5$)	6	7	7	6	5	6	7	5	11	8	8	6
Wa ($r = 7$)	5	6	6	5	4	5	5	4	9	7	6	4
Wa ($r = 9$)	5	5	5	5	3	4	4	3	5	6	5	3
Wa ($r = 11$)	5	6	5	5	3	3	3	2	3	4	4	2
Wa ($r = 13$)	5	6	5	5	3	3	3	2	3	3	3	1
Wa ($r = 15$)	5	6	6	5	3	3	3	2	2	3	2	1

Table 5.13: Over all rankings of each spatial aggregate for each base match cost function at 95% confidence for each data set, Table 2 of 2. Rankings are oriented vertically.

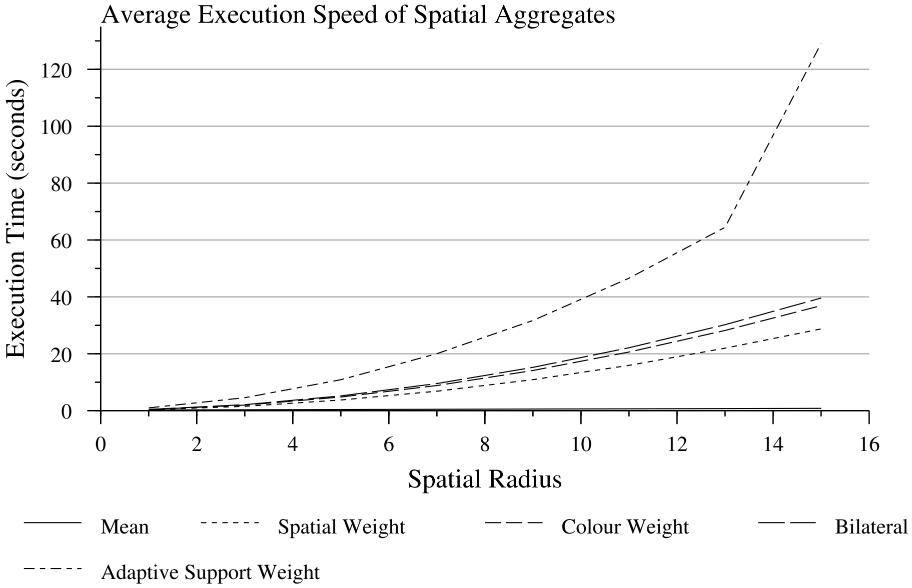
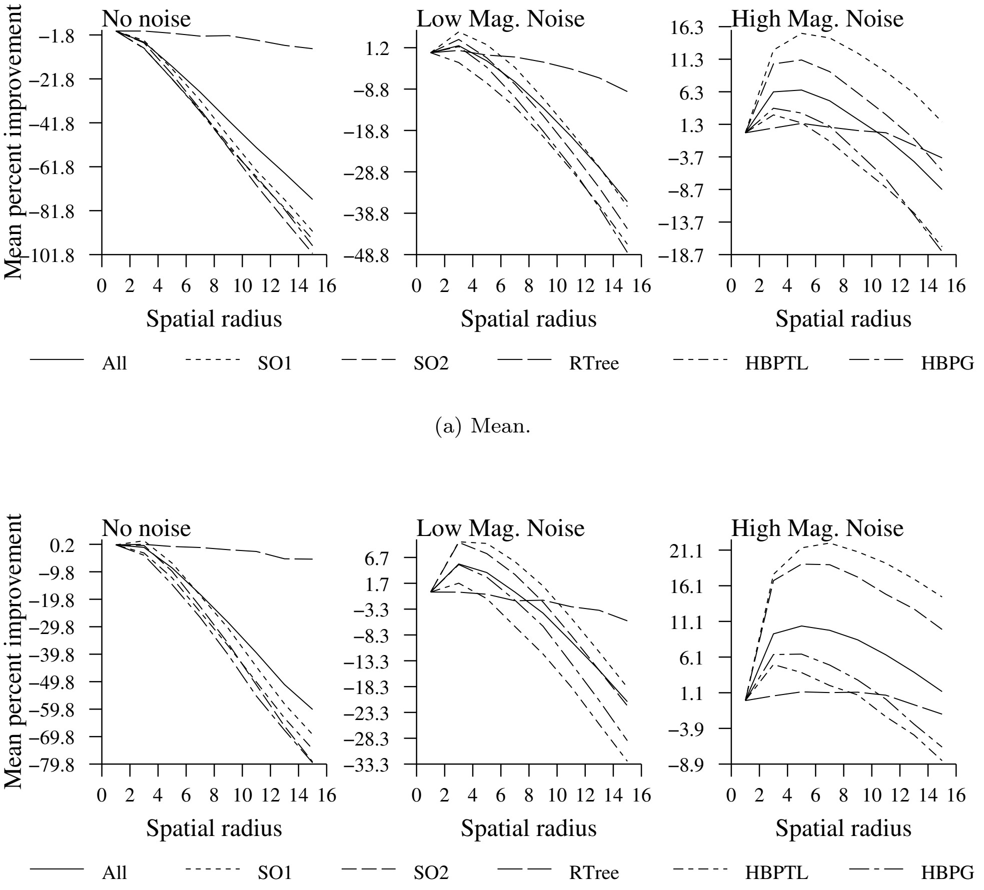


Figure 5.9: Spatial radius versus mean execution time of each spatial aggregate. Mean is calculated from 20 applications of the spatial aggregate, and the execution times were obtained on from a Pentium 4 3.0GHz processor.

the middle 90% of the accuracy improvements taken over all data sets and over all match cost functions that utilize the spatial aggregate to increase robustness to outliers. We also include graphs for each match cost function in Appendix H.

Consulting Table 2.1d (see Section 2.2.2) we see that the spatial aggregates that we include in this study are themselves constructed using up to two weighting functions: one Gaussian weight in the spatial (image pixel) domain, and one Gaussian weight in the spectral (colour) domain. The mean spatial aggregate incorporates neither of these functions; it weighs all pixels in the aggregation window equally. The colour weighted spatial aggregates weighs all pixels in the aggregation window with a Gaussian weight in the spectral domain. The spatial weight employs a Gaussian weight in the spatial domain. The bilateral and adaptive support weight spatial aggregates incorporates both weighting functions; however, the adaptive support weight spatial aggregate calculates the weighting functions centered on both the pixel under consideration and its projection at the desired disparity value. We can use the ranking data in Tables 5.10, 5.11, 5.12, 5.13, and G.1 through G.10 (see Appendix G) to gauge the effectiveness of each of these weighting functions in the context of global stereopsis algorithms.

Comparing the relative rankings of the mean (M) and spatial weight (Ws) spatial aggregates we see that although they do tend to generate statistically significantly similar results at some low values of the spatial radius, r, the spatial weight spatial aggregate consistently produces better results as r is increased. When comparing the rankings of the colour weight (Wc) and bilateral (B) spatial aggregates we see that this same trend holds. This suggests to us that incorporating this spatial weighting term in a spatial aggregate essentially acts as a hedge against choosing too large of a spatial radius, r. That is, at low values of r it does not have much effect, but as r is increased this weighting function reduces the negative impact of incorporating data that is too far from the pixel of interest. This is further reflected in the graphs of Figures 5.10a and 5.10b.



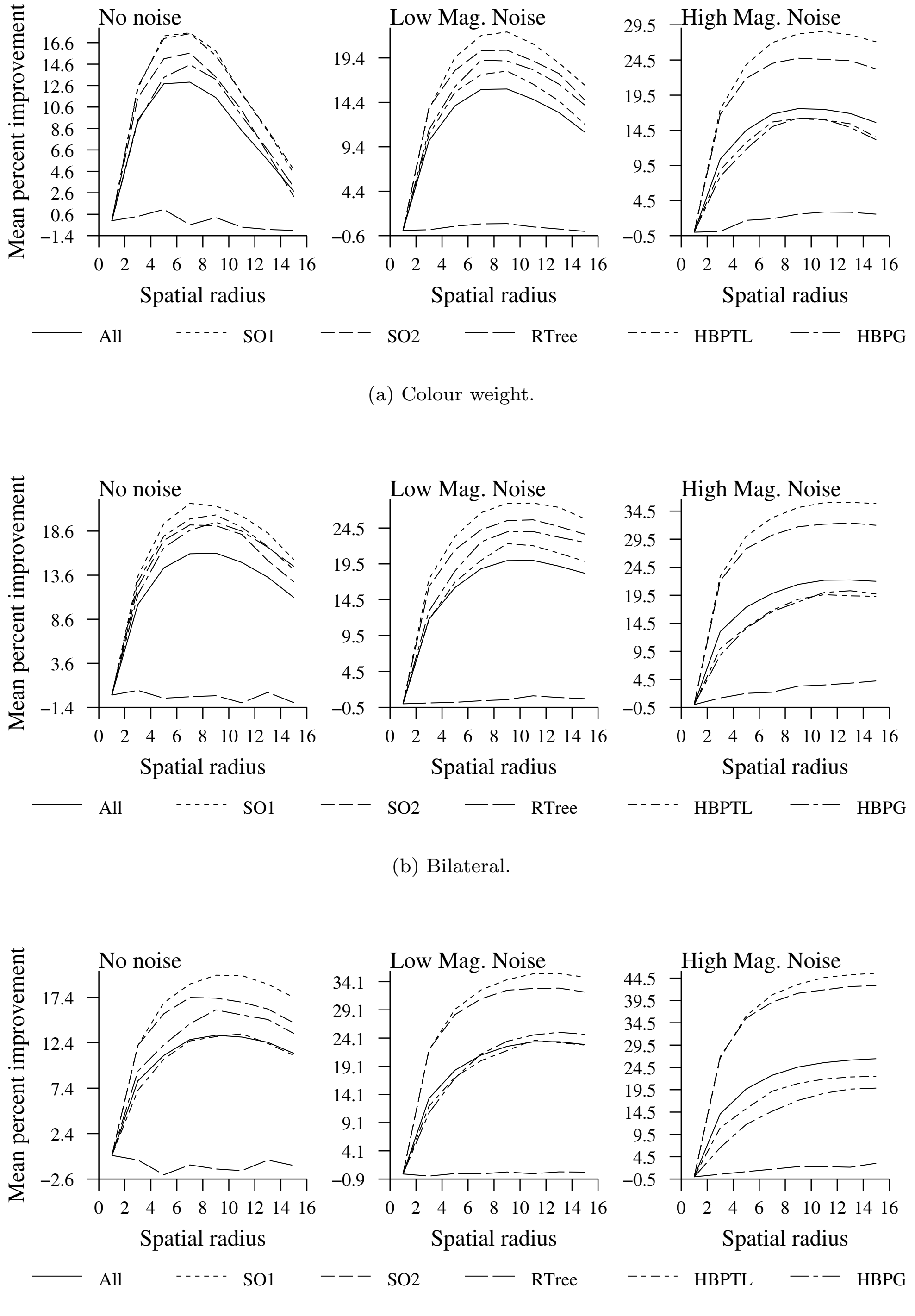
(a) Mean.

(b) Spatial weight.

Figure 5.10: Mean accuracy improvement of the mean and spatial weight spatial aggregates as the spatial radius, r, is increased.

The Gaussian weight in the spectral domain acts as a rudimentary object detector. It assigns a high weight to those pixels that are more likely to be on the same object as the pixel at the center of the aggregation window, by virtue of them having the same/similar colours. If we compare the mean (M) to the colour weight (Wc) spatial aggregate, and the spatial weight (Ws) to the bilateral (B) spatial aggregate then we can make inferences about the effectiveness of the Gaussian weight in the spectral domain as well. Our rankings indicate that except in a few instances at very small spatial radii ($r \leq 3$), the mean and spatial weight spatial aggregates are universally outranked, and thus outperformed, by the colour weight and bilateral spatial aggregates, respectively. Thus, our data indicates that including a Gaussian weight in the spectral domain can almost always be expected to yield a spatial aggregate that is better than one without said weight for global stereopsis algorithms.

Finally, comparing the relative rankings of the bilateral (B) and adaptive support weight (Wa) spatial aggregates we see that the adaptive support weight spatial aggregate universally ranks the same or worse than the bilateral spatial aggregate at the same spatial radius. In fact, the adaptive support weight spatial aggregate typically ranks worse than the bilateral spatial aggregate. Thus, even though a simple winner-take-all algorithm with a match cost function that incorporates the adaptive support weight spatial aggregate can yield very accurate disparity maps [105], our data indicates that this aggregate may not be worth the



(a) Colour weight.

(b) Bilateral.

(c) Adaptive support weight.

Figure 5.11: Mean accuracy improvement of the colour weight, bilateral, and adaptive support weight spatial aggregates as the spatial radius, r, is increased.

substantial implementation complexity and added computational effort for global stereopsis algorithms.

Figure 5.9 presents graphs of the spatial radius versus the mean execution time on a Pentium 4 3.0GHz processor of our spatial aggregates. The data for these graphs was obtained by applying each spatial aggregate 20 times to the SAD match cost function, with 71 disparity values, on the Aloe image pair from our data set. All of these graphs follow fairly clear trends with the exception of the adaptive support weight spatial aggregate. After a spatial radius of 13 (29×29 spatial window), the execution time of this spatial aggregate suddenly leaps. In our implementation of this spatial aggregate, we cache the weights of the warps for a pixel at each disparity value as we traverse the scanlines of the image. This cache allows us to drastically reduce the amount of recalculation that needs to be performed when applying this aggregate. At a spatial radius of 13, this cache fits entirely within the L2 cache of the processor that we use. However, at a spatial radius of 15, this cache no longer fits within the L2 cache of the processor.

5.5 Over All Analysis

In this section we combine and analyze the data from both phases of our study on match cost functions for global stereopsis. In this analysis we consider the aggregate rankings the 304 match cost functions as well as the relative accuracy of the 1520 global stereopsis algorithms constructed, and used, in this study.

5.5.1 Match Cost Function Analysis

In Tables 5.14 through 5.24 we present the ranking of all 304 match cost functions used in this phase of our study. These tables contain the over all and per-framework ranking of the match cost functions on each of our three data sets.

An interesting trend to be observed in this ranking data is that as the magnitude of the noise is increased, the variability in the results obtained by match cost functions within the SO1 and SO2 frameworks increases faster than within the RTree, HBPTL, and HBPG frameworks. That is, the number of unique ranks increases at a faster rate for these two frameworks as noise magnitude is increased. This suggests that the disparity smoothness assumptions made in the SO1 and SO2 frameworks, and expressed in their smoothing prior terms, is too weak to overcome the increased noise variance, and forces the algorithms from these two frameworks to become more dependent on their match cost function. Similarly, the smoothness assumptions made in the HBPTL and HBPG framework appear to be quite strong in the presence of even high magnitude noise.

Segmentation based formulations of stereopsis, such as the RTree framework, tend to heavily rely on the quality of the segmentation used. If a poor segmentation is used, then the accuracy of the formulation will suffer. The ranking data in these tables indicates that this reliance on the quality of the segmentation is severe enough to make it more or less irrelevant which match cost function is used in the RTree formulation.

In our ranking of match cost functions, match cost functions that incorporate a spatial aggregate in their definition exclusively occupy the lowest ranks. On the data set with no noise the lowest over all ranking match cost functions with no spatial aggregate are the MxG[t]D and MxA[t]D functions with a rank of 21. The MxG[t]D function is the lowest over all ranked match cost function with no spatial aggregate on the data set with low magnitude noise at a rank of 26. Finally, the SwS[t]D, Sw[t]SD, and SwSD functions are the lowest over all ranked match cost functions with no spatial aggregate on the data set with high magnitude noise with a rank of 24.

	No noise						Low magnitude noise						High magnitude noise					
Measure	**All**	**SO1**	**SO2**	**RTree**	**HBPTL**	**HBPG**	**All**	**SO1**	**SO2**	**RTree**	**HBPTL**	**HBPG**	**All**	**SO1**	**SO2**	**RTree**	**HBPTL**	**HBPG**
BMxA[t]D ($r = 9$)	1	1	1	1	1	1	2	1	1	1	1	1	4	2	2	1	2	2
BMxA[t]D ($r = 11$)	1	1	1	1	1	1	1	1	1	1	1	1	2	2	2	1	1	1
BMxG[t]D ($r = 9$)	1	1	1	1	1	1	2	2	2	1	1	1	4	2	3	1	2	2
BMxA[t]D ($r = 7$)	1	1	1	1	2	2	3	2	2	1	1	2	5	3	4	1	2	2
BMxG[t]D ($r = 11$)	2	2	1	1	1	1	1	1	1	1	1	1	3	2	2	1	1	1
BSq[t]GD ($r = 9$)	2	1	1	1	1	1	3	2	2	1	1	2	6	4	4	1	3	3
BSq[t]GD ($r = 11$)	2	1	1	1	1	1	2	2	1	1	1	1	5	3	3	1	2	2
WcMxA[t]D ($r = 9$)	2	1	1	1	2	1	1	1	1	1	1	1	4	2	3	1	2	2
WcMxA[t]D ($r = 7$)	2	1	1	1	1	2	2	2	2	1	1	1	4	3	3	1	2	2
BMxG[t]D ($r = 7$)	2	1	1	1	1	1	3	3	2	1	2	1	6	4	4	1	3	3
WcMxG[t]D ($r = 7$)	2	1	2	1	1	1	3	2	2	1	2	2	6	4	3	1	2	2
BMxA[t]D ($r = 13$)	2	1	2	1	2	2	1	1	1	1	1	1	2	1	2	1	1	1
BMxG[t]D ($r = 13$)	2	1	2	1	2	1	1	1	1	1	1	1	2	1	2	1	1	1
WcSq[t]GD ($r = 7$)	3	2	1	1	2	2	4	3	3	1	2	2	7	4	4	1	3	3
WcMxA[t]D ($r = 5$)	3	1	1	1	2	2	4	3	3	1	2	2	7	5	4	1	3	3
WcMxG[t]D ($r = 9$)	3	1	2	1	2	1	2	2	2	1	1	1	4	3	3	1	2	2
WcSq[t]GD ($r = 9$)	3	2	2	1	2	1	3	2	2	1	1	2	6	4	4	1	2	3
BSq[t]GD ($r = 7$)	3	1	2	1	2	2	4	3	3	1	2	2	7	4	5	1	3	4
BSq[t]GD ($r = 13$)	3	2	2	1	2	1	2	1	2	1	1	1	4	3	2	1	2	2
BS[t]AD ($r = 9$)	3	2	1	1	2	2	4	3	2	1	2	2	10	9	5	1	5	5
WcS[t]AD ($r = 7$)	3	2	2	1	2	2	5	3	3	1	2	3	10	9	8	1	5	5
BS[t]AD ($r = 11$)	3	2	2	1	2	2	3	2	2	1	2	2	8	8	5	1	5	4
BMxA[t]D ($r = 15$)	3	2	2	1	2	2	1	1	1	1	1	1	1	1	1	1	1	1
BMxG[t]D ($r = 15$)	3	2	2	1	2	2	2	1	1	1	1	1	2	1	2	1	1	1
WcMxG[t]D ($r = 5$)	3	2	2	1	2	2	5	3	4	1	2	2	7	5	4	1	3	3
WcMxA[t]D ($r = 11$)	3	2	2	1	2	2	2	1	1	1	1	1	3	2	2	1	1	1
BS[t]AD ($r = 7$)	3	2	2	1	2	2	5	4	3	1	2	3	11	10	9	1	5	5
BMxG[t]D ($r = 5$)	3	2	2	1	2	2	6	4	4	1	3	3	8	5	5	1	4	3

Table 5.14: Cluster rankings of match cost functions at 95% confidence, Table 1 of 11. Ranking calculated using only the photographic portion of the data sets.

	No noise						Low magnitude noise						High magnitude noise					
Measure	**All**	**SO1**	**SO2**	**RTree**	**HBPTL**	**HBPG**	**All**	**SO1**	**SO2**	**RTree**	**HBPTL**	**HBPG**	**All**	**SO1**	**SO2**	**RTree**	**HBPTL**	**HBPG**
BSq[t]GD ($r = 15$)	4	2	2	1	2	2	2	2	1	1	1	1	4	3	3	1	2	2
BS[t]AD ($r = 13$)	4	2	2	1	2	2	3	2	2	1	2	2	7	4	5	1	5	4
BMxA[t]D ($r = 5$)	4	2	2	1	2	2	5	3	3	1	2	3	7	5	5	1	3	3
WcS[t]AD ($r = 9$)	4	2	2	1	2	3	4	3	2	1	2	2	10	9	8	1	5	5
WcMxG[t]D ($r = 11$)	4	2	2	1	2	2	2	2	1	1	1	1	3	2	2	1	1	1
WcSq[t]GD ($r = 5$)	4	2	2	1	2	2	6	4	4	1	2	3	9	6	5	1	5	4
WcSq[t]GD ($r = 11$)	4	2	2	1	2	2	3	2	2	1	1	2	5	3	3	1	2	2
WcMxA[t]D ($r = 13$)	4	2	2	1	3	2	2	2	1	1	1	1	2	2	2	1	1	1
BS[t]AD ($r = 15$)	4	2	2	1	3	2	3	2	2	1	1	2	7	8	4	1	5	4
WcS[t]AD ($r = 5$)	5	2	2	1	3	3	7	4	4	1	3	3	13	11	9	1	6	5
BSq[t]GD ($r = 5$)	5	2	2	1	3	3	7	4	4	1	3	3	9	6	5	1	5	4
WcS[t]AD ($r = 11$)	5	3	2	1	3	3	4	2	2	1	2	2	8	8	8	1	5	4
BS[t]AD ($r = 5$)	5	3	2	1	3	3	7	5	4	1	3	3	15	12	10	1	6	6
WcMxG[t]D ($r = 13$)	5	3	5	1	3	2	3	2	1	1	1	1	3	2	2	1	1	2
WcSq[t]GD ($r = 13$)	5	3	2	1	3	3	3	3	2	1	2	2	5	4	3	1	2	2
WcMxA[t]D ($r = 3$)	5	3	2	1	3	3	8	6	5	1	3	3	10	10	6	1	3	4
WcMxG[t]D ($r = 3$)	5	3	2	1	3	3	8	6	5	1	4	3	10	10	6	1	4	4
WcS[t]AD ($r = 13$)	6	3	5	1	3	3	4	6	2	1	2	3	9	8	7	1	5	5
WcMxA[t]D ($r = 15$)	6	3	3	1	3	3	3	2	2	1	2	2	3	2	2	1	1	2
WcSq[t]GD ($r = 15$)	6	3	3	1	3	3	5	3	3	1	4	2	6	3	3	1	5	3
WcSq[t]GD ($r = 3$)	6	3	3	1	3	3	9	7	5	1	4	4	11	11	6	1	5	4
BMxA[t]D ($r = 3$)	6	3	3	1	3	3	8	7	5	1	3	3	10	10	9	1	3	4
WcMxG[t]D ($r = 15$)	6	4	5	1	3	3	4	3	3	1	2	2	3	2	2	1	2	2
BMxG[t]D ($r = 3$)	6	3	3	1	3	3	9	7	5	1	4	4	12	11	9	1	5	4
WaMxG[t]D ($r = 11$)	6	3	3	1	3	3	3	2	2	1	2	2	5	3	3	1	2	3
WcS[t]AD ($r = 15$)	7	4	5	1	4	3	6	6	3	1	4	3	10	8	7	1	5	5
WaMxG[t]D ($r = 9$)	8	4	3	1	4	3	4	3	3	1	2	3	6	4	4	1	3	3
WcS[t]AD ($r = 3$)	8	4	3	1	4	3	9	7	5	1	4	4	17	13	11	1	7	6

Table 5.15: Cluster rankings of match cost functions at 95% confidence, Table 2 of 11. Ranking calculated using only the photographic portion of the data sets.

	No noise						Low magnitude noise						High magnitude noise					
Measure	**All**	**SO1**	**SO2**	**RTree**	**HBPTL**	**HBPG**	**All**	**SO1**	**SO2**	**RTree**	**HBPTL**	**HBPG**	**All**	**SO1**	**SO2**	**RTree**	**HBPTL**	**HBPG**
WaMxA[t]D ($r = 9$)	8	4	3	1	4	3	4	2	2	1	2	2	5	3	3	1	2	3
WaMxA[t]D ($r = 11$)	8	4	3	1	4	3	3	2	2	1	2	2	4	3	2	1	2	2
WaMxA[t]D ($r = 13$)	8	4	5	1	4	3	3	2	1	1	2	2	3	2	2	1	2	2
WaMxG[t]D ($r = 13$)	8	4	5	1	4	3	3	2	2	1	2	2	4	2	2	1	2	2
BSq[t]GD ($r = 3$)	8	4	3	1	4	3	9	7	5	1	4	4	12	11	9	1	6	5
WaSq[t]GD ($r = 13$)	8	4	5	1	4	3	3	2	2	1	2	2	5	3	3	1	2	3
WaMxG[t]D ($r = 7$)	8	4	4	1	4	3	6	4	3	1	3	3	8	8	5	1	3	4
WaSq[t]GD ($r = 9$)	8	4	4	1	4	3	5	3	3	1	2	3	6	4	4	1	3	3
WaSq[t]GD ($r = 11$)	8	4	4	1	4	3	4	3	2	1	2	3	6	3	3	1	3	3
WaMxG[t]D ($r = 15$)	8	4	5	1	4	3	3	2	2	1	2	2	3	2	2	1	2	2
WaMxA[t]D ($r = 7$)	8	4	3	1	4	3	5	3	3	1	3	3	7	5	4	1	2	3
WaMxA[t]D ($r = 15$)	8	4	5	1	4	3	3	2	2	1	2	2	3	2	2	1	1	2
WaSq[t]GD ($r = 15$)	8	4	5	1	4	3	4	2	2	1	2	3	5	3	3	1	2	3
WaSq[t]GD ($r = 7$)	9	4	3	1	4	3	6	4	3	1	3	3	8	8	5	1	3	4
BS[t]AD ($r = 3$)	9	4	4	1	4	4	10	7	6	1	4	5	18	14	12	1	7	7
WaS[t]AD ($r = 13$)	9	5	5	1	5	3	5	3	2	1	2	3	10	9	8	1	5	5
WaMxA[t]D ($r = 5$)	9	5	4	1	5	4	7	5	4	1	4	4	10	10	8	1	5	4
WaS[t]AD ($r = 11$)	9	5	5	1	4	4	5	3	3	1	2	3	10	9	8	1	5	5
WaMxG[t]D ($r = 5$)	9	5	4	1	5	3	8	6	4	1	4	4	11	11	8	1	5	4
WaS[t]AD ($r = 9$)	9	4	4	1	5	4	6	3	3	1	3	3	14	10	9	1	6	6
WaS[t]AD ($r = 7$)	10	5	4	1	4	4	7	4	4	1	3	4	15	12	10	1	6	6
WaSq[t]GD ($r = 5$)	10	5	4	1	5	4	8	6	4	1	4	4	11	11	8	1	5	5
WaS[t]AD ($r = 5$)	11	5	6	1	5	4	9	6	7	1	4	5	18	14	15	1	7	7
WaMxA[t]D ($r = 3$)	11	6	5	1	5	4	10	8	8	1	4	5	16	13	11	1	6	5
WaMxG[t]D ($r = 3$)	11	6	6	1	6	5	15	8	8	1	5	5	17	14	15	1	6	6
WaSq[t]GD ($r = 3$)	12	6	6	1	6	4	11	8	8	1	5	5	17	14	11	1	6	6
WcMxG[t]D ($r = 1$)	12	6	6	1	6	5	17	12	10	1	6	6	18	15	16	1	7	6
WcMxA[t]D ($r = 1$)	12	6	6	1	6	5	16	12	9	1	5	5	17	14	12	1	6	5

Table 5.16: Cluster rankings of match cost functions at 95% confidence, Table 3 of 11. Ranking calculated using only the photographic portion of the data sets.

	No noise						Low magnitude noise						High magnitude noise					
Measure	**All**	**SO1**	**SO2**	**RTree**	**HBPTL**	**HBPG**	**All**	**SO1**	**SO2**	**RTree**	**HBPTL**	**HBPG**	**All**	**SO1**	**SO2**	**RTree**	**HBPTL**	**HBPG**
WaS[t]AD ($r = 3$)	12	7	6	1	6	5	16	9	9	1	5	6	23	17	20	1	10	8
WcSq[t]GD ($r = 1$)	12	6	6	1	6	4	17	12	9	1	5	6	18	14	15	1	7	6
BMxA[t]D ($r = 1$)	13	7	7	1	7	5	18	13	10	1	5	6	20	16	17	1	6	6
WsMxA[t]D ($r = 1$)	13	7	7	1	7	5	17	12	10	1	5	5	18	15	13	1	6	5
BSq[t]GD ($r = 1$)	13	7	7	1	7	5	18	13	11	1	6	7	20	16	17	1	10	7
WsMxG[t]D ($r = 1$)	13	7	7	1	7	5	18	13	11	1	6	6	19	15	16	1	7	6
WsMxG[t]D ($r = 3$)	14	7	7	1	8	6	11	8	9	1	5	5	12	11	9	1	5	4
WaS[t]AD ($r = 15$)	15	5	5	1	5	4	5	3	2	1	3	3	10	8	7	1	5	5
WcS[t]AD ($r = 1$)	16	7	7	1	6	5	18	13	11	1	6	6	25	20	25	1	11	8
WsSq[t]GD ($r = 1$)	17	9	7	1	7	5	18	12	10	1	5	6	18	15	16	1	7	6
BMxG[t]D ($r = 1$)	17	7	7	1	7	6	19	13	11	1	6	7	21	17	17	1	10	7
MMxA[t]D ($r = 1$)	17	9	7	1	8	6	16	12	9	1	5	5	16	13	11	1	6	5
WsS[t]AD ($r = 1$)	18	10	7	1	7	5	19	13	12	1	6	6	26	21	26	1	11	8
MSq[t]GD ($r = 1$)	18	9	7	1	8	5	17	12	10	1	5	5	17	14	11	1	7	6
WsMxA[t]D ($r = 3$)	18	7	7	1	11	6	11	8	6	1	5	5	11	11	9	1	5	4
MMxG[t]D ($r = 1$)	18	7	7	1	10	6	18	12	10	1	6	5	17	14	11	1	6	6
BS[t]AD ($r = 1$)	18	10	7	1	7	6	21	14	12	1	9	7	29	25	31	1	11	8
WsSq[t]GD ($r = 3$)	18	7	8	1	11	6	11	9	6	1	5	5	13	11	10	1	6	5
MS[t]AD ($r = 1$)	18	10	7	1	8	6	18	13	10	1	6	6	22	18	18	1	11	7
WsS[t]AD ($r = 3$)	19	9	8	1	11	6	12	9	9	1	6	5	18	14	12	1	7	6
WaMxA[t]D ($r = 1$)	19	10	9	1	7	6	21	16	13	1	6	7	26	24	27	1	10	7
WaMxG[t]D ($r = 1$)	19	10	9	1	8	7	22	17	13	1	10	7	25	22	21	1	11	7
WaSq[t]GD ($r = 1$)	19	10	9	1	8	7	22	16	13	1	10	8	25	21	21	1	11	7
MMxA[t]D ($r = 3$)	20	10	9	1	9	7	12	8	9	1	5	5	11	10	6	1	5	4
MMxG[t]D ($r = 3$)	20	10	8	1	12	7	16	11	9	1	6	5	12	11	9	1	6	4
WaS[t]AD ($r = 1$)	20	12	9	1	8	7	31	21	21	1	10	8	39	38	37	1	12	9
MSq[t]GD ($r = 3$)	20	10	9	1	12	7	13	9	9	1	6	5	13	11	10	1	6	5
MS[t]AD ($r = 3$)	21	11	9	1	12	8	17	10	10	1	6	6	17	14	11	1	8	6

Table 5.17: Cluster rankings of match cost functions at 95% confidence, Table 4 of 11. Ranking calculated using only the photographic portion of the data sets.

	No noise						Low magnitude noise						High magnitude noise					
Measure	**All**	**SO1**	**SO2**	**RTree**	**HBPTL**	**HBPG**	**All**	**SO1**	**SO2**	**RTree**	**HBPTL**	**HBPG**	**All**	**SO1**	**SO2**	**RTree**	**HBPTL**	**HBPG**
MxG[t]D	21	11	11	1	8	7	26	21	19	1	10	8	27	26	23	1	11	7
MxA[t]D	21	12	11	1	8	7	29	21	16	1	9	8	29	28	32	1	8	7
WsMxA[t]D ($r = 5$)	21	11	9	1	9	8	11	8	6	1	5	5	10	9	6	1	5	4
WsMxG[t]D ($r = 5$)	21	8	10	1	9	7	12	8	9	1	6	6	11	10	6	1	6	4
Sq[t]GD	21	12	11	1	12	8	29	21	19	1	10	8	27	27	26	1	11	8
Sq[t]SD	22	12	12	1	9	8	30	21	20	1	10	7	36	34	36	1	11	7
Sq[t]G[t2]D	22	12	11	1	12	8	29	18	19	1	10	8	27	25	26	1	11	8
Sq[t]S[t2]D	22	12	11	1	12	8	30	19	20	1	7	8	36	35	36	1	11	7
WsSq[t]GD ($r = 5$)	22	11	11	1	13	8	13	9	10	1	6	6	12	10	7	1	7	5
Sq[t]LD	22	13	12	1	12	7	30	19	16	1	10	8	27	27	26	1	11	8
SqG[t]D	22	12	12	1	12	8	29	21	19	1	10	8	26	25	22	1	11	8
Sq[t]L[t2]D	23	13	12	1	12	8	30	21	16	1	10	8	28	27	26	1	11	8
S[t]GD	23	13	12	1	9	8	30	21	16	1	11	8	28	28	25	1	12	9
MxGD	23	12	12	1	12	9	30	18	16	1	11	8	26	25	22	1	11	7
S[t]G[t2]D	23	13	12	1	12	8	30	21	19	1	11	8	27	26	25	1	11	8
WsS[t]AD ($r = 5$)	23	12	11	1	13	8	14	9	10	1	6	6	16	12	11	1	7	6
S[t]AD	23	13	12	1	12	8	36	25	26	1	10	8	40	41	38	1	15	9
SA[t]D	23	13	12	1	12	8	30	22	19	1	10	8	30	28	28	1	11	8
SqA[t]D	23	13	12	1	12	8	32	23	21	1	10	8	33	32	33	1	11	8
SG[t]D	23	12	12	1	12	9	30	21	19	1	11	8	27	26	25	1	11	8
Sw[t]AD	23	13	12	2	12	8	29	21	16	1	10	8	26	24	26	1	11	8
MxL[t]D	23	12	13	1	12	8	30	21	19	1	11	8	28	27	26	1	12	8
S[t]A[t2]D	23	13	12	1	11	8	36	27	26	1	11	8	40	41	38	1	15	9
SwA[t]D	23	13	12	1	12	8	30	20	16	1	11	8	26	22	24	1	11	8
Sw[t]A[t2]D	23	13	12	1	12	8	30	18	19	1	11	8	27	25	27	1	11	8
SqL[t]D	23	13	12	1	12	9	30	21	16	1	10	8	26	27	25	1	11	8
Sw[t]G[t2]D	23	13	12	1	12	9	30	21	16	1	11	9	25	24	20	1	12	8
SqGD	23	13	12	1	13	9	30	21	17	1	11	8	27	25	23	1	12	8

Table 5.18: Cluster rankings of match cost functions at 95% confidence, Table 5 of 11. Ranking calculated using only the photographic portion of the data sets.

	No noise						Low magnitude noise						High magnitude noise					
Measure	**All**	**SO1**	**SO2**	**RTree**	**HBPTL**	**HBPG**	**All**	**SO1**	**SO2**	**RTree**	**HBPTL**	**HBPG**	**All**	**SO1**	**SO2**	**RTree**	**HBPTL**	**HBPG**
Sw[t]GD	24	13	13	1	13	9	30	21	15	1	11	9	25	21	20	1	12	8
SqS[t]D	24	12	12	1	12	10	29	21	16	1	10	8	26	25	25	1	11	8
SwG[t]D	24	13	12	1	12	9	30	21	15	1	11	9	25	21	20	1	11	8
SqAD	24	13	12	1	12	9	31	23	20	1	10	8	29	28	26	1	11	8
S[t]LD	24	13	14	1	13	8	30	21	19	1	11	8	27	27	25	1	12	8
MxAD	24	12	12	1	16	10	29	18	16	1	10	8	26	25	21	1	11	8
SL[t]D	24	13	14	1	12	9	30	21	16	1	11	9	28	28	23	1	11	8
SGD	25	13	13	1	13	9	31	22	19	1	12	9	28	28	25	1	12	9
S[t]L[t2]D	25	13	14	1	13	8	30	21	19	1	11	8	28	26	25	1	11	9
SqLD	25	13	12	1	13	10	30	21	19	1	11	8	27	25	25	1	11	8
MxLD	25	13	11	1	14	10	31	22	19	1	11	8	28	26	23	1	12	8
SwGD	25	14	13	1	16	9	31	21	16	1	12	9	26	22	20	1	12	9
Sw[t]L[t2]D	25	14	15	1	13	9	30	21	16	1	11	9	25	21	20	1	12	8
SqSD	26	13	13	1	13	10	30	18	19	1	10	9	26	23	22	1	11	8
Sw[t]LD	26	14	15	1	13	9	30	21	15	1	11	9	25	21	20	1	11	8
MMxG[t]D ($r = 5$)	26	13	14	1	14	9	17	9	10	1	7	6	11	10	9	1	6	5
S[t]SD	26	14	14	1	13	9	30	21	16	1	11	8	27	26	22	1	11	8
SwAD	26	13	13	1	16	10	30	21	16	1	11	9	25	22	21	1	11	8
MMxA[t]D ($r = 5$)	27	13	14	1	14	9	13	9	9	1	6	6	10	9	6	1	6	4
SwL[t]D	27	14	15	1	14	9	30	18	16	1	11	9	25	21	20	1	12	9
S[t]S[t2]D	27	13	14	1	13	9	30	21	19	1	11	8	27	26	25	1	11	8
MSq[t]GD ($r = 5$)	27	13	14	1	14	9	18	12	11	1	7	6	13	11	10	1	7	5
WsMxG[t]D ($r = 7$)	27	13	14	1	14	9	17	9	9	1	7	6	11	6	6	1	6	5
WsMxA[t]D ($r = 7$)	27	13	14	1	14	9	17	9	9	1	6	6	10	9	6	1	6	4
WsSq[t]GD ($r = 7$)	28	13	15	1	14	9	18	10	10	1	7	6	13	10	7	1	7	5
MS[t]AD ($r = 5$)	28	14	15	1	17	9	19	12	11	1	10	6	17	13	11	1	7	7
MxS[t]D	28	14	13	1	16	11	30	21	19	1	10	8	27	26	22	1	11	8
WsS[t]AD ($r = 7$)	28	13	15	1	14	9	19	10	11	1	7	6	16	12	11	1	8	6

Table 5.19: Cluster rankings of match cost functions at 95% confidence, Table 6 of 11. Ranking calculated using only the photographic portion of the data sets.

	No noise						Low magnitude noise						High magnitude noise					
Measure	**All**	**SO1**	**SO2**	**RTree**	**HBPTL**	**HBPG**	**All**	**SO1**	**SO2**	**RTree**	**HBPTL**	**HBPG**	**All**	**SO1**	**SO2**	**RTree**	**HBPTL**	**HBPG**
SLD	29	13	13	1	18	11	31	21	19	2	12	9	28	27	25	1	12	9
SS[t]D	30	14	15	1	18	11	30	21	16	1	10	9	27	26	25	1	11	8
Sw[t]S[t2]D	30	15	15	1	18	11	29	17	15	1	11	8	25	21	20	1	11	8
SwS[t]D	30	15	15	1	18	11	29	17	15	1	11	9	24	21	19	1	9	8
Sw[t]SD	30	15	15	1	18	11	29	18	15	1	11	9	24	20	18	1	11	8
WsMxG[t]D ($r = 9$)	30	15	16	1	14	10	19	12	11	1	8	7	11	10	9	1	6	5
WsSq[t]GD ($r = 9$)	30	15	16	1	18	10	20	12	11	1	8	7	13	11	10	1	7	6
MxSD	31	14	16	1	21	11	31	21	19	1	12	9	29	26	23	1	12	10
MMxG[t]D ($r = 7$)	31	15	16	1	18	11	20	13	11	1	8	7	12	10	9	1	6	5
MMxA[t]D ($r = 7$)	31	15	16	1	18	10	19	13	11	1	10	7	11	10	8	1	6	5
MSq[t]GD ($r = 7$)	32	16	17	1	18	11	21	13	12	1	10	7	15	11	10	1	7	6
SwSD	33	15	16	2	21	12	30	21	15	2	12	9	24	19	20	2	11	8
Sx[t]AD	34	15	14	1	12	9	35	24	25	1	12	10	40	40	38	2	15	10
Sx[t]A[t2]D	34	15	14	1	13	9	35	27	25	1	12	10	40	40	38	1	15	11
MxA[t]B	34	14	14	1	13	9	31	23	19	1	10	8	25	25	21	1	10	7
Sx[t]GD	34	15	14	1	13	9	33	23	20	1	12	10	29	28	25	1	14	10
MxG[t]B	34	15	15	1	13	8	31	23	20	1	11	8	26	26	22	1	11	7
SxA[t]D	34	15	14	1	13	9	33	23	21	1	12	9	37	34	34	1	13	10
SxG[t]D	34	15	14	1	13	10	33	23	20	1	12	10	30	32	26	1	13	9
SAD	34	14	13	1	16	10	31	22	19	1	11	9	27	28	25	1	11	8
Sx[t]G[t2]D	34	15	15	1	13	9	33	23	20	1	13	10	29	28	26	1	12	10
Sq[t]GB	34	15	15	1	14	9	31	23	19	1	11	8	26	26	22	1	9	7
Sq[t]SB	34	14	16	1	16	9	31	23	19	1	10	8	25	25	21	1	9	7
Sq[t]S[t2]B	34	15	16	1	16	9	31	23	19	1	10	8	25	25	21	1	9	7
S[t]GB	34	15	16	1	14	9	32	24	20	1	11	8	28	28	23	1	11	7
Sq[t]LB	34	15	16	1	16	9	31	23	19	1	11	8	26	26	21	1	9	7
S[t]AB	34	15	16	1	14	9	31	23	16	1	10	8	27	28	22	1	11	7
Sq[t]G[t2]B	35	15	16	1	14	9	31	23	19	1	11	8	26	26	21	1	11	7

Table 5.20: Cluster rankings of match cost functions at 95% confidence, Table 7 of 11. Ranking calculated using only the photographic portion of the data sets.

	No noise						Low magnitude noise						High magnitude noise					
Measure	**All**	**SO1**	**SO2**	**RTree**	**HBPTL**	**HBPG**	**All**	**SO1**	**SO2**	**RTree**	**HBPTL**	**HBPG**	**All**	**SO1**	**SO2**	**RTree**	**HBPTL**	**HBPG**
SqG[t]B	35	15	16	1	16	9	31	23	20	1	11	8	26	25	21	1	9	7
S[t]G[t2]B	35	15	16	1	16	9	32	24	20	1	11	8	28	29	22	1	11	7
SG[t]B	35	15	16	1	14	9	32	24	20	1	11	8	27	26	22	1	11	8
SA[t]B	35	15	16	1	16	9	31	23	19	1	11	8	27	26	22	1	11	7
S[t]A[t2]B	35	15	16	1	14	9	31	24	19	1	11	8	27	28	25	1	11	7
Sq[t]L[t2]B	35	14	16	1	16	9	32	23	20	1	12	8	26	26	21	1	11	7
SwA[t]B	35	15	16	1	16	9	31	23	19	1	11	8	25	22	18	1	11	7
SqA[t]B	35	15	16	1	16	9	31	23	20	1	10	8	26	25	22	1	10	7
Sw[t]GB	35	15	16	1	16	9	32	23	19	1	12	8	25	25	21	1	11	8
S[t]LB	35	15	16	1	14	9	33	25	20	1	11	8	28	29	23	1	11	7
SxGD	35	15	14	1	16	10	34	23	21	1	12	10	30	29	26	1	13	10
MxGB	35	15	16	1	14	9	32	23	20	1	11	8	26	25	22	1	11	7
Sw[t]AB	35	15	16	1	16	9	31	23	16	1	11	8	25	22	20	1	11	7
SwG[t]B	35	15	16	1	16	9	32	23	20	1	12	8	25	25	20	1	11	7
SqGB	35	15	16	1	16	9	31	23	20	1	11	8	25	25	21	1	11	7
Sw[t]G[t2]B	35	15	16	1	16	9	32	23	20	1	12	8	26	25	21	1	11	8
SqL[t]B	35	15	16	1	16	10	32	23	20	1	11	8	26	25	21	1	9	8
SqAB	36	15	16	1	16	9	31	23	19	1	11	8	25	25	21	1	11	7
Sw[t]A[t2]B	36	15	16	1	17	10	31	23	19	1	11	8	25	25	18	1	11	7
S[t]L[t2]B	36	15	16	1	16	9	33	25	20	1	11	8	29	30	23	1	11	8
SwL[t]B	36	15	16	1	16	9	33	25	20	1	12	8	27	26	22	1	12	8
SGB	36	15	16	1	16	9	32	23	20	1	11	8	27	28	22	1	11	8
MxL[t]B	36	15	16	1	16	9	33	25	20	1	12	8	28	29	22	1	12	8
MxAB	36	15	15	1	17	10	32	23	19	1	12	8	27	25	21	1	12	8
SwLD	36	14	14	1	17	11	30	18	19	1	11	9	25	21	20	1	11	8
SqS[t]B	36	15	16	1	17	10	32	23	19	1	12	8	26	25	21	1	12	8
SL[t]B	36	15	16	1	16	9	33	25	20	1	12	8	29	32	23	1	12	8
Sx[t]LD	36	15	16	1	14	9	33	23	20	1	12	10	30	29	26	1	12	9

Table 5.21: Cluster rankings of match cost functions at 95% confidence, Table 8 of 11. Ranking calculated using only the photographic portion of the data sets.

	No noise						Low magnitude noise						High magnitude noise					
Measure	**All**	**SO1**	**SO2**	**RTree**	**HBPTL**	**HBPG**	**All**	**SO1**	**SO2**	**RTree**	**HBPTL**	**HBPG**	**All**	**SO1**	**SO2**	**RTree**	**HBPTL**	**HBPG**
SwGB	36	15	16	1	17	9	32	23	20	1	12	8	26	22	21	1	12	8
Sw[t]LB	36	15	16	2	16	9	33	25	20	1	12	8	27	26	22	1	12	8
SwAB	36	15	16	1	17	10	32	23	19	1	12	8	26	22	20	1	12	8
SqSB	36	15	16	1	17	10	32	23	19	1	12	9	27	25	21	1	12	8
Sw[t]L[t2]B	37	15	16	1	17	9	33	25	20	1	12	8	27	26	21	1	12	8
SqLB	37	15	16	1	17	10	32	23	20	1	12	8	27	25	20	1	12	8
SxAD	37	15	15	1	18	11	33	23	21	1	12	10	36	33	27	1	14	10
SAB	37	15	16	1	17	10	33	23	19	2	13	9	28	26	21	1	12	9
Sx[t]L[t2]D	37	15	16	1	17	10	33	23	20	1	12	10	30	29	26	1	12	9
SxLD	37	15	15	1	17	12	33	23	20	1	12	10	30	28	26	1	12	9
SxL[t]D	37	16	16	1	14	11	33	23	20	1	12	10	30	29	26	1	12	9
MxS[t]B	37	15	17	2	17	10	34	25	20	1	13	9	30	29	23	1	13	10
S[t]S[t2]B	38	16	17	1	17	10	33	25	20	1	12	9	30	29	22	1	13	9
Sq[t]AB	38	17	17	1	16	9	36	29	26	1	10	8	36	37	35	1	11	7
Sq[t]A[t2]B	38	17	17	1	16	9	36	29	26	1	10	8	36	37	35	1	11	7
S[t]SB	38	16	17	1	17	10	34	25	20	1	13	9	30	30	22	1	13	10
WsMxA[t]D ($r = 9$)	38	15	16	1	15	10	18	12	10	1	7	6	10	7	6	1	6	4
Sw[t]S[t2]B	39	16	17	2	17	11	34	25	20	1	13	9	28	26	21	1	13	9
Sw[t]SB	39	16	17	2	18	10	34	27	20	2	13	9	27	26	21	1	12	9
SwS[t]B	39	16	17	2	18	11	34	25	20	2	13	10	28	26	21	1	13	9
SS[t]B	39	16	17	2	18	10	34	25	20	2	13	9	31	30	22	2	15	10
Sx[t]AB	39	17	17	1	17	10	35	28	21	1	12	9	34	35	27	2	13	10
WsS[t]AD ($r = 9$)	39	15	16	1	18	11	21	13	12	1	8	7	17	12	11	1	8	7
Sx[t]G[t2]B	39	17	17	1	18	9	36	29	24	1	13	9	37	36	27	1	15	10
MxLB	39	16	17	2	21	10	34	25	20	1	13	9	30	29	23	1	13	9
Sx[t]GB	39	17	19	1	17	10	36	28	22	1	13	9	37	35	27	1	15	10
SxA[t]B	40	17	17	1	18	10	35	28	21	1	12	9	34	35	26	1	13	9
Sx[t]S[t2]D	40	16	16	1	18	12	32	23	20	1	11	9	29	28	26	1	12	9

Table 5.22: Cluster rankings of match cost functions at 95% confidence, Table 9 of 11. Ranking calculated using only the photographic portion of the data sets.

	No noise						Low magnitude noise						High magnitude noise					
Measure	**All**	**SO1**	**SO2**	**RTree**	**HBPTL**	**HBPG**	**All**	**SO1**	**SO2**	**RTree**	**HBPTL**	**HBPG**	**All**	**SO1**	**SO2**	**RTree**	**HBPTL**	**HBPG**
MS[t]AD ($r = 7$)	40	16	16	1	18	11	21	13	12	1	11	8	17	12	11	1	8	7
SxG[t]B	40	17	17	1	18	10	35	28	24	1	13	9	37	36	30	1	13	10
SLB	40	16	17	3	18	11	35	26	21	2	13	10	30	30	23	2	15	9
Sx[t]SD	40	16	16	1	18	12	32	23	20	1	11	9	29	28	25	1	13	9
SSD	40	15	16	2	21	13	31	21	19	2	12	9	27	26	22	1	12	9
Sx[t]LB	40	17	18	1	18	10	37	32	24	1	13	9	35	39	29	1	15	10
Sx[t]L[t2]B	40	17	19	1	18	10	37	30	24	1	14	9	38	37	30	2	15	10
SxS[t]D	40	16	17	1	21	13	32	22	20	1	12	10	29	28	25	1	13	9
WsMxG[t]D ($r = 11$)	41	17	18	1	19	11	21	13	13	1	10	8	13	10	9	1	7	6
WsMxA[t]D ($r = 11$)	41	17	18	1	19	11	21	13	12	1	10	7	12	10	9	1	6	6
MxSB	42	16	19	3	21	13	35	25	21	2	14	10	31	29	23	2	16	10
WsSq[t]GD ($r = 11$)	42	17	18	1	19	12	22	14	13	1	11	8	16	11	10	1	8	6
SxSD	42	17	17	2	21	13	33	23	20	2	13	10	29	28	26	2	12	9
WsS[t]AD ($r = 11$)	42	17	18	1	20	12	22	14	13	1	11	8	17	12	15	1	9	7
MMxA[t]D ($r = 9$)	42	17	18	1	20	12	22	14	13	1	11	8	13	10	9	1	7	6
MMxG[t]D ($r = 9$)	43	17	19	1	20	12	22	14	13	1	12	8	15	11	10	1	7	6
MSq[t]GD ($r = 9$)	43	17	18	1	20	12	23	15	14	1	12	8	17	12	11	1	10	6
MS[t]AD ($r = 9$)	44	18	19	1	21	12	24	15	14	1	12	8	18	13	15	1	11	7
WsMxA[t]D ($r = 13$)	45	18	20	1	21	12	23	14	13	1	12	8	13	10	10	1	7	6
WsMxG[t]D ($r = 13$)	45	18	20	1	21	12	24	15	14	1	12	8	15	11	10	1	7	6
WsSq[t]GD ($r = 13$)	46	18	20	1	21	13	25	16	15	1	12	9	17	12	14	1	10	7
WsS[t]AD ($r = 13$)	47	18	20	1	22	13	26	16	18	1	12	9	19	13	15	1	11	8
MMxA[t]D ($r = 11$)	48	19	20	1	23	13	26	16	18	1	12	9	16	11	14	1	10	6
MMxG[t]D ($r = 11$)	48	19	21	1	23	13	30	16	18	1	13	9	16	11	14	1	10	7
MSq[t]GD ($r = 11$)	48	19	21	1	23	13	27	17	18	1	13	9	18	12	15	1	11	7
WsMxA[t]D ($r = 15$)	49	19	21	1	23	13	27	16	18	1	13	9	16	11	14	1	10	7
WsMxG[t]D ($r = 15$)	49	19	21	1	23	13	27	17	19	1	13	10	16	11	14	1	10	7
WsSq[t]GD ($r = 15$)	49	20	21	1	23	14	28	20	19	1	13	10	18	12	15	1	11	7

Table 5.23: Cluster rankings of match cost functions at 95% confidence, Table 10 of 11. Ranking calculated using only the photographic portion of the data sets.

	No noise						Low magnitude noise						High magnitude noise					
Measure	**All**	**SO1**	**SO2**	**RTree**	**HBPTL**	**HBPG**	**All**	**SO1**	**SO2**	**RTree**	**HBPTL**	**HBPG**	**All**	**SO1**	**SO2**	**RTree**	**HBPTL**	**HBPG**
WsS[t]AD ($r = 15$)	50	21	22	1	24	14	31	20	19	1	13	10	20	13	16	1	11	8
Sx[t]A[t2]B	51	17	17	1	18	10	35	28	23	1	13	9	34	35	29	1	13	10
SxL[t]B	52	17	19	1	18	10	37	29	24	1	14	10	38	39	29	2	15	10
SwLB	52	17	17	2	18	12	35	25	20	1	13	10	29	27	22	2	13	9
SxGB	52	17	19	1	18	10	36	28	24	1	13	9	37	35	30	1	13	9
SxAB	53	17	18	1	21	11	36	28	21	2	14	9	37	35	26	2	16	10
Sx[t]SB	54	17	19	2	18	11	37	30	24	2	14	10	38	37	30	2	15	10
SxS[t]B	55	18	20	2	21	11	38	32	24	2	14	10	39	39	30	2	16	10
Sx[t]S[t2]B	55	18	20	2	21	11	37	30	24	2	14	10	38	37	30	2	15	10
SSB	55	17	19	3	18	13	36	27	21	2	14	10	32	31	23	2	16	10
SwSB	56	17	19	3	18	13	35	27	21	1	13	10	29	27	22	2	13	10
Sq[t]AD	57	19	20	1	21	11	41	33	27	1	16	11	41	42	38	1	18	12
Sq[t]A[t2]D	57	18	20	2	21	11	41	33	27	1	16	12	42	42	38	1	18	12
SxLB	57	18	20	2	18	13	39	32	25	2	15	10	39	37	30	2	17	10
SxSB	57	18	20	3	21	15	40	31	25	2	15	11	39	37	30	3	17	11
MS[t]AD ($r = 11$)	57	20	21	1	23	13	31	20	19	1	13	9	20	13	16	1	11	8
MMxA[t]D ($r = 13$)	58	22	23	1	25	15	32	21	19	1	13	10	18	12	15	1	10	7
MMxG[t]D ($r = 13$)	58	22	23	1	25	15	32	21	19	2	13	10	18	12	15	1	11	7
MSq[t]GD ($r = 13$)	59	23	23	1	26	15	33	21	20	1	14	10	19	13	16	1	11	8
MS[t]AD ($r = 13$)	60	23	24	1	27	15	33	21	20	1	14	10	21	14	16	1	14	9
MMxA[t]D ($r = 15$)	61	24	25	1	28	16	34	23	20	1	15	11	20	13	16	1	11	8
MMxG[t]D ($r = 15$)	61	24	25	1	28	16	35	23	23	1	15	11	20	14	16	1	13	8
MSq[t]GD ($r = 15$)	62	24	25	1	29	16	35	23	23	1	15	11	21	14	17	1	14	9
MS[t]AD ($r = 15$)	63	24	26	1	30	16	35	23	23	1	15	11	22	15	19	1	14	10

Table 5.24: Cluster rankings of match cost functions at 95% confidence, Table 11 of 11. Ranking calculated using only the photographic portion of the data sets.

Algorithm	Algorithm
BMxA[t]D ($r = 11$) + HBPTL	BMxA[t]D ($r = 11$) + HBPG
BMxG[t]D ($r = 11$) + HBPTL	BMxG[t]D ($r = 11$) + HBPG
BMxA[t]D ($r = 13$) + SO1	BMxA[t]D ($r = 13$) + SO2
BMxG[t]D ($r = 13$) + SO1	BMxG[t]D ($r = 13$) + SO2
BMxG[t]D ($r = 13$) + HBPTL	BMxG[t]D ($r = 13$) + HBPG
BMxG[t]D ($r = 15$) + HBPTL	

Table 5.25: Algorithms that ranked one on all three data sets.

5.5.2 Relative Algorithm Performance

Tables 5.26, 5.27, and 5.28 present the top 300 global stereopsis algorithms, out of 1520 algorithms, from our study on the data set with no noise introduced as obtained by applying our cluster ranking method at 95% confidence. Within this ranking, algorithms with the same rank are sorted in ascending order by their mean rank from the scenes in this data set. Tables 5.29, 5.30, and 5.31 present the same data, but for the data set with low magnitude noise, and Tables 5.32, 5.33, and 5.34 present the same for the data set with high magnitude noise.

Table 5.25 shows the 11 algorithms that obtained a rank of one on all three of our data sets. All 11 of these algorithms utilize either the BMxA[t]D or BMxG[t]D match cost function with a global stereopsis framework other than the RTree framework. Although not shown in any of these ranking tables, algorithms based on the RTree framework tended to yield the lowest accuracy disparity maps amongst our algorithms.

5.6 Summary

In this Chapter we present the results from, and analysis for, our study on match cost functions for global stereopsis algorithms. For this study we use a large set of constructable match cost functions, that are assembled in a structured manner, within five different global stereopsis frameworks. The structured nature of constructable match cost functions allows us to to systematically analyze the relative performance of the components that comprise these match cost functions as well as the relative performance of the match cost functions themselves.

We divide our analysis into two phases: the first considers 144 match cost functions that do not include some sort of spatial aggregate, and the second considers 160 match cost functions that do include a spatial aggregate.

Our analysis indicates that it is unlikely that a match cost function that can be considered a “Swiss army knife” exists. That is, there does not appear to be any one match cost function that is perfect for any situation. The different properties of global stereopsis algorithms, within which a match cost function can be used, differ in such ways that the ideal match cost function must be chosen specifically for that algorithm.

That said, our analysis does indicate that all is not lost: some general guidelines can be formed for choosing a decent match cost function for global stereopsis algorithms. Specifically:

- The choice of match cost function used in a segmentation based formulation of stereopsis is likely to be irrelevant;
- Making a match cost function more robust to sensor noise and outliers by truncating it, in some manner, will almost always result in a better match cost function;

Algorithm	Rank	Algorithm	Rank
BMxA[t]D ($r = 11$) + SO2	1	WcMxG[t]D ($r = 5$) + SO1	1
BSq[t]GD ($r = 11$) + SO2	1	WcMxG[t]D ($r = 5$) + SO2	1
BMxA[t]D ($r = 9$) + SO2	1	BSq[t]GD ($r = 13$) + HBPG	1
BMxA[t]D ($r = 7$) + SO2	1	BS[t]AD ($r = 13$) + SO2	1
BMxG[t]D ($r = 11$) + HBPG	1	WcSq[t]GD ($r = 9$) + SO2	1
BMxG[t]D ($r = 11$) + HBPTL	1	BS[t]AD ($r = 7$) + SO1	1
BSq[t]GD ($r = 9$) + SO2	1	BMxA[t]D ($r = 7$) + HBPG	1
WcMxA[t]D ($r = 7$) + SO1	1	BMxG[t]D ($r = 11$) + SO1	1
BMxA[t]D ($r = 9$) + SO1	1	BMxG[t]D ($r = 15$) + HBPTL	1
BMxA[t]D ($r = 9$) + HBPTL	1	BSq[t]GD ($r = 11$) + HBPG	1
BMxA[t]D ($r = 11$) + HBPTL	1	BMxA[t]D ($r = 13$) + SO1	1
BMxG[t]D ($r = 9$) + HBPG	1	BMxG[t]D ($r = 13$) + SO2	1
BMxA[t]D ($r = 11$) + SO1	1	BMxA[t]D ($r = 13$) + HBPG	2
BMxG[t]D ($r = 9$) + HBPTL	1	WcMxA[t]D ($r = 9$) + HBPTL	2
BMxA[t]D ($r = 7$) + SO1	1	WcSq[t]GD ($r = 7$) + HBPG	2
BMxA[t]D ($r = 9$) + HBPG	1	WcS[t]AD ($r = 7$) + SO1	2
WcMxA[t]D ($r = 7$) + SO2	1	BS[t]AD ($r = 9$) + SO1	2
WcMxA[t]D ($r = 9$) + SO2	1	WcS[t]AD ($r = 7$) + SO2	2
BMxG[t]D ($r = 9$) + SO2	1	WcMxG[t]D ($r = 9$) + SO2	2
BMxA[t]D ($r = 11$) + HBPG	1	BMxA[t]D ($r = 15$) + SO2	2
BMxG[t]D ($r = 13$) + HBPTL	1	WcSq[t]GD ($r = 7$) + HBPTL	2
BSq[t]GD ($r = 9$) + HBPG	1	BS[t]AD ($r = 7$) + SO2	2
BMxG[t]D ($r = 13$) + HBPG	1	BSq[t]GD ($r = 13$) + SO2	2
BMxG[t]D ($r = 7$) + SO1	1	WcMxG[t]D ($r = 9$) + HBPTL	2
BMxG[t]D ($r = 9$) + SO1	1	BMxA[t]D ($r = 13$) + HBPTL	2
BMxG[t]D ($r = 11$) + SO2	1	WcMxA[t]D ($r = 7$) + HBPG	2
WcMxG[t]D ($r = 7$) + HBPTL	1	BMxG[t]D ($r = 15$) + HBPG	2
BMxA[t]D ($r = 7$) + HBPTL	1	BSq[t]GD ($r = 13$) + HBPTL	2
WcMxA[t]D ($r = 9$) + SO1	1	WcS[t]AD ($r = 7$) + HBPTL	2
WcMxG[t]D ($r = 7$) + SO1	1	BMxA[t]D ($r = 5$) + SO1	2
WcMxA[t]D ($r = 5$) + SO2	1	WcMxG[t]D ($r = 9$) + HBPG	2
WcSq[t]GD ($r = 7$) + SO2	1	BMxA[t]D ($r = 5$) + SO2	2
BMxG[t]D ($r = 7$) + HBPG	1	BS[t]AD ($r = 11$) + SO2	2
BMxG[t]D ($r = 7$) + SO2	1	WcSq[t]GD ($r = 9$) + HBPTL	2
BSq[t]GD ($r = 7$) + SO1	1	BMxA[t]D ($r = 15$) + HBPG	2
WcMxA[t]D ($r = 7$) + HBPTL	1	BSq[t]GD ($r = 15$) + HBPG	2
BMxA[t]D ($r = 13$) + SO2	1	BS[t]AD ($r = 11$) + SO1	2
BSq[t]GD ($r = 9$) + HBPTL	1	BMxG[t]D ($r = 15$) + SO2	2
WcMxA[t]D ($r = 5$) + SO1	1	WcMxA[t]D ($r = 5$) + HBPTL	2
BMxG[t]D ($r = 13$) + SO1	1	WcS[t]AD ($r = 9$) + SO2	2
BSq[t]GD ($r = 7$) + SO2	1	BMxG[t]D ($r = 5$) + SO2	2
WcMxA[t]D ($r = 11$) + SO2	1	WcMxA[t]D ($r = 9$) + HBPG	2
BMxG[t]D ($r = 7$) + HBPTL	1	WcSq[t]GD ($r = 9$) + HBPG	2
BSq[t]GD ($r = 11$) + HBPTL	1	BSq[t]GD ($r = 7$) + HBPTL	2
WcMxG[t]D ($r = 7$) + SO2	1	BS[t]AD ($r = 9$) + HBPTL	2
WcSq[t]GD ($r = 7$) + SO1	1	BMxA[t]D ($r = 15$) + HBPTL	2
BSq[t]GD ($r = 9$) + SO1	1	BSq[t]GD ($r = 13$) + SO1	2
BS[t]AD ($r = 9$) + SO2	1	BS[t]AD ($r = 11$) + HBPTL	2
WcMxG[t]D ($r = 7$) + HBPG	1	BMxA[t]D ($r = 15$) + SO1	2
BSq[t]GD ($r = 11$) + SO1	1	WcMxA[t]D ($r = 11$) + SO1	2

Table 5.26: 300 top ranked algorithms on the data set with no noise, Table 1 of 3. Rankings calculated with cluster ranking at 95% confidence.

Algorithm	**Rank**	**Algorithm**	**Rank**
WcMxA[t]D ($r = 13$) + SO2	2	WcMxG[t]D ($r = 11$) + SO1	4
WcMxG[t]D ($r = 11$) + HBPG	2	WcSq[t]GD ($r = 5$) + HBPTL	4
BMxG[t]D ($r = 5$) + SO1	2	WcSq[t]GD ($r = 11$) + SO2	4
WcSq[t]GD ($r = 5$) + SO2	2	BS[t]AD ($r = 7$) + HBPTL	4
WcMxG[t]D ($r = 5$) + HBPG	2	WcS[t]AD ($r = 9$) + SO1	4
BSq[t]GD ($r = 7$) + HBPG	2	BSq[t]GD ($r = 15$) + SO1	4
WcS[t]AD ($r = 5$) + SO2	2	WcS[t]AD ($r = 9$) + HBPTL	4
BMxG[t]D ($r = 5$) + HBPTL	2	BSq[t]GD ($r = 15$) + HBPTL	4
BMxG[t]D ($r = 5$) + HBPG	2	WcMxA[t]D ($r = 11$) + HBPG	4
WcMxG[t]D ($r = 5$) + HBPTL	2	BS[t]AD ($r = 13$) + HBPG	4
WcSq[t]GD ($r = 5$) + SO1	2	WcSq[t]GD ($r = 11$) + SO1	4
WcMxG[t]D ($r = 11$) + HBPTL	2	WcS[t]AD ($r = 5$) + HBPTL	4
BMxA[t]D ($r = 5$) + HBPTL	2	BS[t]AD ($r = 15$) + SO1	4
BS[t]AD ($r = 11$) + HBPG	2	WcS[t]AD ($r = 11$) + SO1	4
BS[t]AD ($r = 9$) + HBPG	2	WcSq[t]GD ($r = 13$) + SO2	4
WcMxA[t]D ($r = 5$) + HBPG	2	WcSq[t]GD ($r = 11$) + HBPG	4
BMxG[t]D ($r = 15$) + SO1	2	WcS[t]AD ($r = 9$) + HBPG	4
BS[t]AD ($r = 5$) + SO2	2	WcMxG[t]D ($r = 13$) + HBPG	4
BSq[t]GD ($r = 5$) + SO2	2	BS[t]AD ($r = 15$) + HBPTL	4
WcS[t]AD ($r = 5$) + SO1	2	WcMxA[t]D ($r = 13$) + SO1	4
BMxA[t]D ($r = 5$) + HBPG	2	WcMxA[t]D ($r = 13$) + HBPTL	4
BS[t]AD ($r = 7$) + HBPG	2	BS[t]AD ($r = 15$) + HBPG	4
WcMxA[t]D ($r = 3$) + SO2	2	WcSq[t]GD ($r = 11$) + HBPTL	4
WcS[t]AD ($r = 7$) + HBPG	2	WcMxG[t]D ($r = 13$) + HBPTL	4
BSq[t]GD ($r = 5$) + SO1	2	WcSq[t]GD ($r = 5$) + HBPG	4
BSq[t]GD ($r = 5$) + HBPG	2	WcMxA[t]D ($r = 13$) + HBPG	4
BS[t]AD ($r = 5$) + SO1	2	WcS[t]AD ($r = 13$) + SO2	4
BSq[t]GD ($r = 5$) + HBPTL	2	WcMxG[t]D ($r = 13$) + SO2	4
BS[t]AD ($r = 5$) + HBPTL	2	WcS[t]AD ($r = 11$) + HBPTL	4
WcMxG[t]D ($r = 3$) + SO2	2	WcSq[t]GD ($r = 13$) + HBPTL	4
WcMxA[t]D ($r = 3$) + SO1	2	WcS[t]AD ($r = 5$) + HBPG	4
WcMxG[t]D ($r = 3$) + SO1	2	BMxA[t]D ($r = 3$) + SO2	4
WcS[t]AD ($r = 3$) + SO2	2	WcMxG[t]D ($r = 13$) + SO1	4
WcSq[t]GD ($r = 3$) + SO2	2	WcSq[t]GD ($r = 15$) + SO2	4
WaMxG[t]D ($r = 13$) + SO2	2	WcSq[t]GD ($r = 13$) + SO1	4
BMxA[t]D ($r = 3$) + HBPTL	2	WcMxA[t]D ($r = 15$) + SO2	5
BMxG[t]D ($r = 3$) + SO2	2	WcS[t]AD ($r = 13$) + SO1	5
WaMxG[t]D ($r = 11$) + SO2	3	BS[t]AD ($r = 5$) + HBPG	5
WaMxG[t]D ($r = 9$) + SO2	3	WcMxG[t]D ($r = 15$) + SO2	5
WaMxG[t]D ($r = 7$) + SO2	3	WcS[t]AD ($r = 11$) + HBPG	5
WaMxA[t]D ($r = 5$) + SO2	3	WcMxA[t]D ($r = 3$) + HBPTL	5
WcMxG[t]D ($r = 9$) + SO1	4	WcMxG[t]D ($r = 3$) + HBPTL	5
WcSq[t]GD ($r = 9$) + SO1	4	WcS[t]AD ($r = 15$) + SO2	5
BS[t]AD ($r = 15$) + SO2	4	WcMxG[t]D ($r = 3$) + HBPG	5
WcMxA[t]D ($r = 11$) + HBPTL	4	WcSq[t]GD ($r = 13$) + HBPG	5
BSq[t]GD ($r = 15$) + SO2	4	WcMxA[t]D ($r = 15$) + HBPG	5
WcMxG[t]D ($r = 11$) + SO2	4	WcMxA[t]D ($r = 3$) + HBPG	5
BS[t]AD ($r = 13$) + SO1	4	WcMxG[t]D ($r = 15$) + HBPTL	5
WcS[t]AD ($r = 11$) + SO2	4	WcSq[t]GD ($r = 15$) + HBPG	5
BS[t]AD ($r = 13$) + HBPTL	4	WcMxG[t]D ($r = 15$) + HBPG	5

Table 5.27: 300 top ranked algorithms on the data set with no noise, Table 2 of 3. Rankings calculated with cluster ranking at 95% confidence.

Algorithm	**Rank**	**Algorithm**	**Rank**
WaMxG[t]D ($r = 9$) + HBPG	5	WaMxG[t]D ($r = 13$) + HBPTL	5
WcS[t]AD ($r = 13$) + HBPG	5	WaMxA[t]D ($r = 13$) + SO1	5
WaMxG[t]D ($r = 15$) + HBPG	5	WaSq[t]GD ($r = 11$) + HBPG	5
BMxG[t]D ($r = 3$) + HBPTL	5	WaMxA[t]D ($r = 15$) + HBPG	5
BMxA[t]D ($r = 3$) + HBPG	5	WaSq[t]GD ($r = 13$) + SO1	5
WcSq[t]GD ($r = 3$) + HBPTL	5	WaMxA[t]D ($r = 7$) + HBPG	5
WcMxA[t]D ($r = 15$) + SO1	5	WaS[t]AD ($r = 11$) + SO2	5
WaMxA[t]D ($r = 15$) + SO2	5	WcS[t]AD ($r = 15$) + HBPTL	5
WaMxA[t]D ($r = 7$) + SO2	5	WaMxA[t]D ($r = 7$) + SO1	5
WcMxA[t]D ($r = 15$) + HBPTL	5	WaSq[t]GD ($r = 9$) + SO1	5
WaMxA[t]D ($r = 9$) + SO2	5	WaMxA[t]D ($r = 15$) + SO1	5
WaSq[t]GD ($r = 11$) + SO2	5	WaMxG[t]D ($r = 7$) + SO1	5
WcSq[t]GD ($r = 15$) + HBPTL	5	WaSq[t]GD ($r = 11$) + SO1	5
WcS[t]AD ($r = 13$) + HBPTL	5	WaMxG[t]D ($r = 15$) + SO1	5
WaMxA[t]D ($r = 9$) + HBPG	5	WaS[t]AD ($r = 13$) + SO2	5
WaMxG[t]D ($r = 13$) + HBPG	5	WaSq[t]GD ($r = 7$) + HBPG	5
WcS[t]AD ($r = 15$) + SO1	5	BSq[t]GD ($r = 3$) + SO1	5
BSq[t]GD ($r = 3$) + SO2	5	WaMxA[t]D ($r = 11$) + HBPTL	5
WaSq[t]GD ($r = 13$) + HBPG	5	WcS[t]AD ($r = 3$) + SO1	5
WcSq[t]GD ($r = 15$) + SO1	5	WaMxG[t]D ($r = 7$) + HBPTL	5
WaMxG[t]D ($r = 11$) + SO1	5	WaMxG[t]D ($r = 5$) + SO2	5
BSq[t]GD ($r = 3$) + HBPG	5	WaS[t]AD ($r = 9$) + SO2	5
BMxG[t]D ($r = 3$) + SO1	5	WcS[t]AD ($r = 3$) + HBPG	5
WaSq[t]GD ($r = 15$) + SO2	5	WaSq[t]GD ($r = 7$) + SO1	5
WaMxG[t]D ($r = 7$) + HBPG	5	WaS[t]AD ($r = 7$) + SO2	5
WaMxA[t]D ($r = 13$) + SO2	5	WaMxA[t]D ($r = 13$) + HBPTL	5
WaMxA[t]D ($r = 13$) + HBPG	5	WcS[t]AD ($r = 3$) + HBPTL	5
BS[t]AD ($r = 3$) + SO2	5	WaMxA[t]D ($r = 9$) + HBPTL	5
WaMxA[t]D ($r = 11$) + SO2	5	WaS[t]AD ($r = 15$) + SO2	5
WaSq[t]GD ($r = 13$) + SO2	5	BS[t]AD ($r = 3$) + HBPTL	5
WaMxG[t]D ($r = 11$) + HBPG	5	BSq[t]GD ($r = 3$) + HBPTL	5
BMxG[t]D ($r = 3$) + HBPG	5	WaSq[t]GD ($r = 15$) + SO1	5
WaSq[t]GD ($r = 9$) + SO2	5	BS[t]AD ($r = 3$) + HBPG	5
WaMxG[t]D ($r = 15$) + SO2	5	WaMxA[t]D ($r = 15$) + HBPTL	5
WcSq[t]GD ($r = 3$) + SO1	5	WaSq[t]GD ($r = 11$) + HBPTL	5
WcMxG[t]D ($r = 15$) + SO1	5	WaS[t]AD ($r = 9$) + SO1	5
BMxA[t]D ($r = 3$) + SO1	5	WaSq[t]GD ($r = 5$) + SO2	5
WaMxA[t]D ($r = 11$) + SO1	5	WaMxA[t]D ($r = 7$) + HBPTL	5
WaMxG[t]D ($r = 11$) + HBPTL	5	WaSq[t]GD ($r = 15$) + HBPTL	5
WaMxG[t]D ($r = 9$) + SO1	5	WaMxA[t]D ($r = 5$) + HBPG	5
WaMxG[t]D ($r = 9$) + HBPTL	5	BS[t]AD ($r = 3$) + SO1	5
WcSq[t]GD ($r = 3$) + HBPG	5	WaS[t]AD ($r = 13$) + SO1	5
WcS[t]AD ($r = 15$) + HBPG	5	WaMxG[t]D ($r = 5$) + HBPG	5
WaSq[t]GD ($r = 7$) + SO2	5	WaS[t]AD ($r = 9$) + HBPG	5
WaMxG[t]D ($r = 15$) + HBPTL	5	WaSq[t]GD ($r = 13$) + HBPTL	5
WaMxA[t]D ($r = 11$) + HBPG	5	WaSq[t]GD ($r = 7$) + HBPTL	5
WaSq[t]GD ($r = 15$) + HBPG	5	WaS[t]AD ($r = 11$) + SO1	5
WaSq[t]GD ($r = 9$) + HBPG	5	WaSq[t]GD ($r = 9$) + HBPTL	6
WaMxG[t]D ($r = 13$) + SO1	5	WaS[t]AD ($r = 13$) + HBPG	6
WaMxA[t]D ($r = 9$) + SO1	5	WaMxG[t]D ($r = 5$) + SO1	6

Table 5.28: 300 top ranked algorithms on the data set with no noise, Table 3 of 3. Rankings calculated with cluster ranking at 95% confidence.

Algorithm	**Rank**	**Algorithm**	**Rank**
BMxA[t]D ($r = 13$) + SO2	1	WcMxA[t]D ($r = 13$) + HBPTL	1
BMxA[t]D ($r = 11$) + HBPTL	1	WcSq[t]GD ($r = 9$) + HBPG	1
BMxG[t]D ($r = 13$) + HBPTL	1	WcMxA[t]D ($r = 13$) + HBPG	1
BMxA[t]D ($r = 13$) + HBPTL	1	WcMxG[t]D ($r = 9$) + HBPTL	1
BMxG[t]D ($r = 11$) + HBPTL	1	BSq[t]GD ($r = 15$) + HBPTL	1
BMxA[t]D ($r = 11$) + HBPG	1	WcMxA[t]D ($r = 13$) + SO1	1
BMxA[t]D ($r = 15$) + SO2	1	BSq[t]GD ($r = 13$) + SO1	1
BMxG[t]D ($r = 15$) + HBPTL	1	WcMxG[t]D ($r = 11$) + SO2	1
WcMxA[t]D ($r = 9$) + HBPTL	1	WcMxG[t]D ($r = 7$) + HBPTL	1
BMxA[t]D ($r = 15$) + HBPG	1	BMxG[t]D ($r = 9$) + SO2	1
BMxA[t]D ($r = 9$) + HBPTL	1	BS[t]AD ($r = 13$) + SO2	1
BMxA[t]D ($r = 13$) + SO1	1	BSq[t]GD ($r = 13$) + SO2	1
BMxG[t]D ($r = 15$) + SO2	1	BSq[t]GD ($r = 13$) + HBPG	2
BMxA[t]D ($r = 15$) + HBPTL	1	BS[t]AD ($r = 13$) + HBPTL	2
BMxA[t]D ($r = 11$) + SO2	1	WcMxG[t]D ($r = 9$) + SO1	2
BMxA[t]D ($r = 9$) + HBPG	1	BSq[t]GD ($r = 11$) + SO2	2
BMxA[t]D ($r = 13$) + HBPG	1	BSq[t]GD ($r = 15$) + SO2	2
BMxG[t]D ($r = 11$) + HBPG	1	WaMxG[t]D ($r = 13$) + SO2	2
BMxG[t]D ($r = 13$) + HBPG	1	WcMxG[t]D ($r = 13$) + HBPTL	2
BMxG[t]D ($r = 9$) + HBPG	1	WcMxG[t]D ($r = 11$) + HBPG	2
BMxG[t]D ($r = 15$) + HBPG	1	BS[t]AD ($r = 9$) + HBPTL	2
WcMxA[t]D ($r = 13$) + SO2	1	BSq[t]GD ($r = 9$) + HBPG	2
BMxG[t]D ($r = 9$) + HBPTL	1	WcMxA[t]D ($r = 7$) + SO2	2
BMxG[t]D ($r = 13$) + SO1	1	WaMxA[t]D ($r = 13$) + SO2	2
WcMxA[t]D ($r = 9$) + SO2	1	WcSq[t]GD ($r = 11$) + HBPTL	2
BSq[t]GD ($r = 11$) + HBPTL	1	WaMxA[t]D ($r = 11$) + HBPTL	2
BSq[t]GD ($r = 11$) + HBPG	1	WaMxG[t]D ($r = 11$) + HBPTL	2
WcMxA[t]D ($r = 9$) + HBPG	1	WcSq[t]GD ($r = 9$) + HBPTL	2
BMxG[t]D ($r = 13$) + SO2	1	WcMxG[t]D ($r = 7$) + HBPG	2
BMxG[t]D ($r = 11$) + SO1	1	BMxG[t]D ($r = 7$) + HBPG	2
WcMxA[t]D ($r = 7$) + HBPTL	1	BMxA[t]D ($r = 7$) + HBPG	2
WcMxA[t]D ($r = 9$) + SO1	1	WcSq[t]GD ($r = 11$) + HBPG	2
BMxA[t]D ($r = 15$) + SO1	1	WcMxG[t]D ($r = 13$) + HBPG	2
BMxG[t]D ($r = 11$) + SO2	1	BSq[t]GD ($r = 15$) + SO1	2
WcMxG[t]D ($r = 11$) + HBPTL	1	WaMxA[t]D ($r = 13$) + SO1	2
BSq[t]GD ($r = 9$) + HBPTL	1	WaMxG[t]D ($r = 13$) + SO1	2
WcMxG[t]D ($r = 9$) + HBPG	1	WaMxA[t]D ($r = 13$) + HBPTL	2
WcMxA[t]D ($r = 11$) + SO2	1	WcMxG[t]D ($r = 11$) + SO1	2
WcMxA[t]D ($r = 11$) + HBPG	1	BMxG[t]D ($r = 9$) + SO1	2
WcMxA[t]D ($r = 11$) + HBPTL	1	BS[t]AD ($r = 11$) + SO2	2
BMxA[t]D ($r = 11$) + SO1	1	WaMxA[t]D ($r = 15$) + SO2	2
BMxA[t]D ($r = 9$) + SO1	1	WcSq[t]GD ($r = 7$) + HBPTL	2
BMxA[t]D ($r = 7$) + HBPTL	1	BMxG[t]D ($r = 7$) + HBPTL	2
BSq[t]GD ($r = 13$) + HBPTL	1	WaMxA[t]D ($r = 13$) + HBPG	2
BSq[t]GD ($r = 15$) + HBPG	1	WaSq[t]GD ($r = 13$) + HBPTL	2
WcMxA[t]D ($r = 7$) + HBPG	1	BS[t]AD ($r = 13$) + HBPG	2
BMxG[t]D ($r = 15$) + SO1	1	BS[t]AD ($r = 15$) + HBPTL	2
WcMxA[t]D ($r = 11$) + SO1	1	WcMxG[t]D ($r = 13$) + SO2	2
BMxA[t]D ($r = 9$) + SO2	1	BSq[t]GD ($r = 11$) + SO1	2
BS[t]AD ($r = 11$) + HBPTL	1	WaMxG[t]D ($r = 13$) + HBPG	2

Table 5.29: 300 top ranked algorithms on the data set with low magnitude noise, Table 1 of 3. Rankings calculated with cluster ranking at 95% confidence.

Algorithm	**Rank**	**Algorithm**	**Rank**
WcMxA[t]D ($r = 5$) + HBPG	2	BSq[t]GD ($r = 7$) + HBPTL	2
WaMxG[t]D ($r = 11$) + HBPG	2	WaMxA[t]D ($r = 11$) + HBPG	2
BS[t]AD ($r = 9$) + SO2	2	WaS[t]AD ($r = 15$) + SO1	2
WaMxA[t]D ($r = 11$) + SO1	2	WaSq[t]GD ($r = 15$) + HBPG	2
BS[t]AD ($r = 15$) + SO1	2	WcMxG[t]D ($r = 7$) + SO2	2
WaMxA[t]D ($r = 15$) + SO1	2	WcMxG[t]D ($r = 7$) + SO1	2
WaMxA[t]D ($r = 15$) + HBPG	2	WaMxA[t]D ($r = 7$) + SO2	2
WaMxA[t]D ($r = 11$) + SO2	2	WaSq[t]GD ($r = 11$) + HBPG	2
WaMxG[t]D ($r = 15$) + HBPTL	2	WcMxA[t]D ($r = 5$) + SO2	2
BS[t]AD ($r = 11$) + HBPG	2	WaSq[t]GD ($r = 15$) + HBPTL	2
BS[t]AD ($r = 15$) + HBPG	2	WcSq[t]GD ($r = 9$) + SO1	2
WcSq[t]GD ($r = 9$) + SO2	2	WcSq[t]GD ($r = 11$) + SO1	2
BMxA[t]D ($r = 7$) + SO1	2	BMxG[t]D ($r = 7$) + SO1	2
WaMxG[t]D ($r = 13$) + HBPTL	2	WaMxA[t]D ($r = 9$) + SO1	2
WcMxA[t]D ($r = 7$) + SO1	2	WaSq[t]GD ($r = 11$) + SO1	2
WaSq[t]GD ($r = 15$) + SO1	2	BMxG[t]D ($r = 7$) + SO2	2
WcMxG[t]D ($r = 9$) + SO2	2	WcMxG[t]D ($r = 15$) + HBPG	2
BSq[t]GD ($r = 9$) + SO2	2	WaS[t]AD ($r = 15$) + SO2	2
WaMxG[t]D ($r = 15$) + SO2	2	WaMxG[t]D ($r = 9$) + HBPTL	2
WaSq[t]GD ($r = 13$) + HBPG	2	WaMxG[t]D ($r = 9$) + HBPG	2
BS[t]AD ($r = 13$) + SO1	2	WcSq[t]GD ($r = 13$) + SO1	2
WcMxA[t]D ($r = 15$) + SO1	2	WaSq[t]GD ($r = 9$) + SO1	2
WcMxA[t]D ($r = 15$) + SO2	2	WaSq[t]GD ($r = 9$) + SO2	2
BS[t]AD ($r = 11$) + SO1	2	WaS[t]AD ($r = 11$) + SO2	2
WaMxG[t]D ($r = 11$) + SO2	2	BS[t]AD ($r = 9$) + SO1	2
WaSq[t]GD ($r = 13$) + SO2	2	WcMxG[t]D ($r = 15$) + SO2	2
WaSq[t]GD ($r = 11$) + HBPTL	2	WcS[t]AD ($r = 7$) + HBPTL	2
WcS[t]AD ($r = 11$) + SO2	2	BMxG[t]D ($r = 5$) + HBPG	2
WaMxG[t]D ($r = 15$) + SO1	2	WaS[t]AD ($r = 13$) + SO2	2
BMxA[t]D ($r = 7$) + SO2	2	WcS[t]AD ($r = 11$) + SO1	2
WcSq[t]GD ($r = 7$) + HBPG	2	WaMxG[t]D ($r = 9$) + SO1	2
BSq[t]GD ($r = 7$) + HBPG	2	WaS[t]AD ($r = 13$) + SO1	2
WaMxG[t]D ($r = 11$) + SO1	2	WcS[t]AD ($r = 7$) + SO2	2
BS[t]AD ($r = 15$) + SO2	2	WcSq[t]GD ($r = 7$) + SO2	2
WcSq[t]GD ($r = 13$) + SO2	2	WcMxG[t]D ($r = 5$) + HBPTL	2
WcSq[t]GD ($r = 11$) + SO2	2	WcS[t]AD ($r = 9$) + SO1	2
WcS[t]AD ($r = 9$) + HBPTL	2	WcSq[t]GD ($r = 5$) + HBPTL	2
WaSq[t]GD ($r = 13$) + SO1	2	WcSq[t]GD ($r = 7$) + SO1	2
BSq[t]GD ($r = 9$) + SO1	2	BSq[t]GD ($r = 7$) + SO2	2
WcS[t]AD ($r = 9$) + SO2	2	WaS[t]AD ($r = 11$) + SO1	2
WaSq[t]GD ($r = 15$) + SO2	2	BS[t]AD ($r = 7$) + SO2	2
WaMxA[t]D ($r = 15$) + HBPTL	2	BMxA[t]D ($r = 5$) + SO2	2
WcSq[t]GD ($r = 13$) + HBPG	2	WcMxG[t]D ($r = 5$) + SO1	2
WaMxA[t]D ($r = 9$) + HBPTL	2	BSq[t]GD ($r = 7$) + SO1	2
WcS[t]AD ($r = 11$) + HBPTL	2	WcMxA[t]D ($r = 5$) + SO1	2
WaMxA[t]D ($r = 9$) + SO2	2	WaS[t]AD ($r = 9$) + SO1	2
WcMxA[t]D ($r = 15$) + HBPTL	2	WaS[t]AD ($r = 9$) + SO2	3
WcMxA[t]D ($r = 5$) + HBPTL	2	WcS[t]AD ($r = 7$) + SO1	3
WaMxG[t]D ($r = 9$) + SO2	2	WaMxA[t]D ($r = 7$) + SO1	3
WaSq[t]GD ($r = 11$) + SO2	2	BMxA[t]D ($r = 5$) + SO1	3

Table 5.30: 300 top ranked algorithms on the data set with low magnitude noise, Table 2 of 3. Rankings calculated with cluster ranking at 95% confidence.

Algorithm	Rank	Algorithm	Rank
WaMxG[t]D ($r = 7$) + SO2	3	WaS[t]AD ($r = 13$) + HBPG	6
WaSq[t]GD ($r = 7$) + SO2	3	WaS[t]AD ($r = 15$) + HBPTL	6
BS[t]AD ($r = 7$) + SO1	3	WaSq[t]GD ($r = 9$) + HBPG	6
BMxG[t]D ($r = 5$) + SO1	3	BS[t]AD ($r = 7$) + HBPTL	6
WaMxG[t]D ($r = 7$) + SO1	3	WaS[t]AD ($r = 11$) + HBPG	6
WcMxG[t]D ($r = 5$) + SO2	3	WaS[t]AD ($r = 9$) + HBPTL	6
WaMxA[t]D ($r = 5$) + SO2	3	WcSq[t]GD ($r = 5$) + HBPG	6
BS[t]AD ($r = 5$) + HBPTL	3	WcSq[t]GD ($r = 15$) + SO1	6
WaSq[t]GD ($r = 7$) + SO1	3	WaMxA[t]D ($r = 7$) + HBPTL	6
WcSq[t]GD ($r = 5$) + SO2	3	WaMxA[t]D ($r = 7$) + HBPG	6
BMxG[t]D ($r = 5$) + SO2	3	BS[t]AD ($r = 7$) + HBPG	6
BSq[t]GD ($r = 5$) + SO2	3	WaMxG[t]D ($r = 7$) + HBPG	6
WaS[t]AD ($r = 7$) + SO2	3	WcS[t]AD ($r = 15$) + SO2	6
BS[t]AD ($r = 5$) + SO2	3	WaS[t]AD ($r = 9$) + HBPG	6
WcS[t]AD ($r = 5$) + SO2	3	WaMxG[t]D ($r = 7$) + HBPTL	6
WcSq[t]GD ($r = 5$) + SO1	3	WcMxA[t]D ($r = 3$) + HBPG	6
BSq[t]GD ($r = 5$) + SO1	3	WcS[t]AD ($r = 15$) + HBPG	6
WaMxG[t]D ($r = 5$) + SO2	3	WcS[t]AD ($r = 15$) + HBPTL	6
WaSq[t]GD ($r = 5$) + SO2	3	WcSq[t]GD ($r = 15$) + SO2	6
WcMxA[t]D ($r = 3$) + SO2	4	BMxG[t]D ($r = 5$) + HBPTL	6
BMxA[t]D ($r = 3$) + SO2	4	BS[t]AD ($r = 5$) + HBPG	6
BMxG[t]D ($r = 3$) + SO2	4	BSq[t]GD ($r = 5$) + HBPG	6
WcMxG[t]D ($r = 3$) + SO2	4	WaS[t]AD ($r = 15$) + HBPG	6
WcSq[t]GD ($r = 3$) + SO2	4	WcS[t]AD ($r = 5$) + HBPTL	6
BSq[t]GD ($r = 3$) + SO2	4	WaSq[t]GD ($r = 7$) + HBPG	6
WsMxA[t]D ($r = 3$) + SO2	5	WaSq[t]GD ($r = 7$) + HBPTL	6
WsSq[t]GD ($r = 3$) + SO2	5	WcMxG[t]D ($r = 3$) + HBPG	6
WaMxA[t]D ($r = 9$) + HBPG	6	WcS[t]AD ($r = 15$) + SO1	6
WaMxG[t]D ($r = 15$) + HBPG	6	BSq[t]GD ($r = 5$) + HBPTL	6
WcSq[t]GD ($r = 13$) + HBPTL	6	WcMxA[t]D ($r = 3$) + HBPTL	6
WcS[t]AD ($r = 11$) + HBPG	6	BMxA[t]D ($r = 3$) + HBPG	6
WcMxG[t]D ($r = 13$) + SO1	6	BMxA[t]D ($r = 3$) + HBPTL	6
WcMxG[t]D ($r = 5$) + HBPG	6	WaS[t]AD ($r = 7$) + HBPTL	6
WcMxA[t]D ($r = 15$) + HBPG	6	WcS[t]AD ($r = 5$) + HBPG	6
BS[t]AD ($r = 9$) + HBPG	6	WaS[t]AD ($r = 7$) + SO1	6
WcMxG[t]D ($r = 15$) + HBPTL	6	WcMxG[t]D ($r = 3$) + HBPTL	7
WcS[t]AD ($r = 7$) + HBPG	6	WcS[t]AD ($r = 5$) + SO1	7
WcS[t]AD ($r = 13$) + HBPTL	6	WaMxA[t]D ($r = 5$) + HBPG	7
WcS[t]AD ($r = 13$) + HBPG	6	BS[t]AD ($r = 5$) + SO1	7
WcS[t]AD ($r = 13$) + SO2	6	WaSq[t]GD ($r = 5$) + HBPTL	7
WaS[t]AD ($r = 11$) + HBPTL	6	WaMxA[t]D ($r = 5$) + HBPTL	7
WcSq[t]GD ($r = 15$) + HBPG	6	WcSq[t]GD ($r = 3$) + HBPG	7
WcMxG[t]D ($r = 15$) + SO1	6	WaMxG[t]D ($r = 5$) + HBPTL	7
BMxA[t]D ($r = 5$) + HBPTL	6	WaS[t]AD ($r = 7$) + HBPG	7
WcS[t]AD ($r = 13$) + SO1	6	BMxG[t]D ($r = 3$) + HBPTL	7
WcS[t]AD ($r = 9$) + HBPG	6	BMxG[t]D ($r = 3$) + HBPG	7
WaS[t]AD ($r = 13$) + HBPTL	6	WaMxA[t]D ($r = 5$) + SO1	7
WaSq[t]GD ($r = 9$) + HBPTL	6	WcSq[t]GD ($r = 3$) + HBPTL	7
WcSq[t]GD ($r = 15$) + HBPTL	6	WaS[t]AD ($r = 5$) + HBPTL	7
BMxA[t]D ($r = 5$) + HBPG	6	WaMxG[t]D ($r = 5$) + SO1	7

Table 5.31: 300 top ranked algorithms on the data set with low magnitude noise, Table 3 of 3. Rankings calculated with cluster ranking at 95% confidence.

Algorithm	**Rank**	**Algorithm**	**Rank**
BMxA[t]D ($r = 15$) + HBPG	1	WaMxG[t]D ($r = 15$) + HBPTL	2
BMxA[t]D ($r = 15$) + HBPTL	1	WcMxG[t]D ($r = 13$) + SO1	2
BMxG[t]D ($r = 15$) + HBPTL	1	BMxG[t]D ($r = 9$) + HBPG	2
BMxG[t]D ($r = 13$) + HBPG	1	WaMxA[t]D ($r = 11$) + HBPG	2
BMxA[t]D ($r = 13$) + HBPTL	1	WaMxA[t]D ($r = 11$) + HBPTL	2
BMxA[t]D ($r = 11$) + HBPTL	1	WcMxA[t]D ($r = 11$) + SO1	2
BMxG[t]D ($r = 13$) + HBPTL	1	WcMxG[t]D ($r = 13$) + SO2	2
BMxA[t]D ($r = 15$) + SO1	1	WaMxA[t]D ($r = 13$) + SO2	2
BMxA[t]D ($r = 11$) + HBPG	1	WcMxG[t]D ($r = 15$) + HBPTL	2
BMxA[t]D ($r = 13$) + HBPG	1	WaMxG[t]D ($r = 15$) + HBPG	2
BMxA[t]D ($r = 15$) + SO2	1	BMxA[t]D ($r = 7$) + HBPTL	2
WcMxA[t]D ($r = 11$) + HBPG	1	WcMxA[t]D ($r = 7$) + HBPTL	2
BMxG[t]D ($r = 15$) + HBPG	1	BMxG[t]D ($r = 9$) + HBPTL	2
BMxA[t]D ($r = 13$) + SO2	1	WcMxA[t]D ($r = 7$) + HBPG	2
WcMxA[t]D ($r = 11$) + HBPTL	1	WaMxG[t]D ($r = 15$) + SO1	2
BMxA[t]D ($r = 13$) + SO1	1	WaMxA[t]D ($r = 9$) + HBPTL	2
WcMxA[t]D ($r = 13$) + HBPTL	1	BSq[t]GD ($r = 15$) + HBPTL	2
BMxG[t]D ($r = 11$) + HBPTL	1	WaSq[t]GD ($r = 13$) + SO1	2
BMxG[t]D ($r = 15$) + SO2	1	BSq[t]GD ($r = 11$) + HBPTL	2
WaMxA[t]D ($r = 15$) + HBPTL	1	WcMxG[t]D ($r = 13$) + HBPG	2
BMxG[t]D ($r = 11$) + HBPG	1	BMxA[t]D ($r = 9$) + SO1	2
BMxG[t]D ($r = 13$) + SO1	1	BMxG[t]D ($r = 11$) + SO1	2
BMxG[t]D ($r = 15$) + SO1	1	WcMxG[t]D ($r = 11$) + SO2	2
BMxG[t]D ($r = 13$) + SO2	1	WaMxG[t]D ($r = 11$) + HBPTL	2
WcMxG[t]D ($r = 13$) + HBPTL	1	WaMxG[t]D ($r = 13$) + HBPTL	2
WcMxA[t]D ($r = 13$) + HBPG	2	WcMxG[t]D ($r = 15$) + SO1	2
WcMxA[t]D ($r = 9$) + HBPTL	2	BMxG[t]D ($r = 9$) + SO1	2
WaMxA[t]D ($r = 15$) + SO1	2	WcMxG[t]D ($r = 15$) + SO2	2
WcMxA[t]D ($r = 15$) + HBPTL	2	BSq[t]GD ($r = 13$) + HBPG	2
WcMxG[t]D ($r = 9$) + HBPG	2	BSq[t]GD ($r = 15$) + HBPG	2
WcMxA[t]D ($r = 13$) + SO1	2	WaMxG[t]D ($r = 13$) + HBPG	2
WcMxA[t]D ($r = 13$) + SO2	2	WaMxG[t]D ($r = 13$) + SO1	2
WcMxA[t]D ($r = 11$) + SO2	2	WaSq[t]GD ($r = 15$) + SO1	2
WcMxG[t]D ($r = 9$) + HBPTL	2	BMxG[t]D ($r = 11$) + SO2	2
BMxA[t]D ($r = 9$) + HBPG	2	BMxA[t]D ($r = 7$) + HBPG	2
WaMxA[t]D ($r = 15$) + HBPG	2	BMxA[t]D ($r = 9$) + SO2	2
WaMxA[t]D ($r = 13$) + HBPTL	2	WcMxG[t]D ($r = 9$) + SO2	2
WcMxA[t]D ($r = 9$) + HBPG	2	WcMxA[t]D ($r = 9$) + SO1	2
WcMxG[t]D ($r = 11$) + HBPG	2	WcMxG[t]D ($r = 11$) + SO1	2
WaMxA[t]D ($r = 13$) + SO1	2	WaSq[t]GD ($r = 15$) + HBPTL	2
WcMxA[t]D ($r = 15$) + HBPG	2	BSq[t]GD ($r = 15$) + SO1	2
WaMxG[t]D ($r = 15$) + SO2	2	WaMxA[t]D ($r = 11$) + SO1	2
BMxA[t]D ($r = 9$) + HBPTL	2	WaMxA[t]D ($r = 11$) + SO2	2
WaMxA[t]D ($r = 13$) + HBPG	2	WcMxA[t]D ($r = 9$) + SO2	2
BMxA[t]D ($r = 11$) + SO2	2	WaMxG[t]D ($r = 13$) + SO2	2
WcMxA[t]D ($r = 15$) + SO1	2	WaSq[t]GD ($r = 15$) + SO2	2
BMxA[t]D ($r = 11$) + SO1	2	WaSq[t]GD ($r = 13$) + HBPG	2
WcMxA[t]D ($r = 15$) + SO2	2	BSq[t]GD ($r = 13$) + SO1	2
WcMxG[t]D ($r = 11$) + HBPTL	2	WaSq[t]GD ($r = 13$) + HBPTL	2
WaMxA[t]D ($r = 15$) + SO2	2	WcMxG[t]D ($r = 7$) + HBPTL	2

Table 5.32: 300 top ranked algorithms on the data set with high magnitude noise, Table 1 of 3. Rankings calculated with cluster ranking at 95% confidence.

Algorithm	**Rank**	**Algorithm**	**Rank**
WaMxG[t]D ($r = 11$) + SO2	2	BSq[t]GD ($r = 9$) + SO2	3
WaMxA[t]D ($r = 9$) + HBPG	2	WcSq[t]GD ($r = 9$) + SO1	3
BMxG[t]D ($r = 9$) + SO2	2	WaMxG[t]D ($r = 9$) + SO1	3
BSq[t]GD ($r = 13$) + SO2	2	WaSq[t]GD ($r = 7$) + HBPTL	3
BSq[t]GD ($r = 15$) + SO2	2	WcMxG[t]D ($r = 7$) + SO1	3
WaMxG[t]D ($r = 11$) + HBPG	2	WaMxA[t]D ($r = 7$) + SO2	3
WcMxA[t]D ($r = 7$) + SO1	2	BMxA[t]D ($r = 3$) + HBPTL	3
BSq[t]GD ($r = 13$) + HBPTL	2	WcMxG[t]D ($r = 5$) + SO2	3
WcMxG[t]D ($r = 15$) + HBPG	2	BSq[t]GD ($r = 7$) + SO2	3
BSq[t]GD ($r = 11$) + HBPG	2	WcSq[t]GD ($r = 7$) + SO2	3
WcMxG[t]D ($r = 7$) + HBPG	2	BSq[t]GD ($r = 7$) + SO1	3
WcMxG[t]D ($r = 9$) + SO1	2	WcMxA[t]D ($r = 5$) + SO2	3
WaMxG[t]D ($r = 11$) + SO1	2	BMxG[t]D ($r = 5$) + HBPTL	3
WcMxG[t]D ($r = 7$) + SO2	2	WcMxA[t]D ($r = 3$) + HBPTL	3
WaSq[t]GD ($r = 13$) + SO2	2	BMxA[t]D ($r = 5$) + SO1	4
WcSq[t]GD ($r = 11$) + HBPTL	2	BMxA[t]D ($r = 5$) + SO2	4
WcMxA[t]D ($r = 5$) + HBPG	3	WaMxG[t]D ($r = 7$) + SO2	4
WaMxG[t]D ($r = 9$) + HBPTL	3	WcMxG[t]D ($r = 3$) + HBPTL	4
BMxG[t]D ($r = 7$) + HBPG	3	WcSq[t]GD ($r = 7$) + SO1	4
BSq[t]GD ($r = 11$) + SO1	3	BMxG[t]D ($r = 5$) + SO2	4
WaMxG[t]D ($r = 9$) + HBPG	3	WcSq[t]GD ($r = 5$) + SO2	4
WaSq[t]GD ($r = 15$) + HBPG	3	WsMxA[t]D ($r = 3$) + HBPTL	4
WaMxA[t]D ($r = 9$) + SO2	3	BSq[t]GD ($r = 5$) + SO2	4
WaMxA[t]D ($r = 7$) + HBPTL	3	WsMxA[t]D ($r = 5$) + HBPTL	4
WcSq[t]GD ($r = 9$) + HBPTL	3	WsMxA[t]D ($r = 7$) + SO2	4
WcMxA[t]D ($r = 7$) + SO2	3	WcSq[t]GD ($r = 13$) + HBPTL	5
BMxA[t]D ($r = 5$) + HBPTL	3	WcSq[t]GD ($r = 13$) + HBPG	5
WaSq[t]GD ($r = 11$) + HBPTL	3	WcSq[t]GD ($r = 11$) + HBPG	5
BSq[t]GD ($r = 9$) + HBPG	3	WcSq[t]GD ($r = 7$) + HBPG	5
BMxA[t]D ($r = 7$) + SO1	3	WcSq[t]GD ($r = 9$) + HBPG	5
WaSq[t]GD ($r = 11$) + SO1	3	WcSq[t]GD ($r = 13$) + SO2	5
BMxG[t]D ($r = 7$) + HBPTL	3	WcSq[t]GD ($r = 7$) + HBPTL	5
BMxA[t]D ($r = 5$) + HBPG	3	WcSq[t]GD ($r = 15$) + SO2	5
WcMxA[t]D ($r = 5$) + HBPTL	3	WcSq[t]GD ($r = 15$) + HBPTL	5
WaMxG[t]D ($r = 9$) + SO2	3	WcSq[t]GD ($r = 15$) + SO1	5
BSq[t]GD ($r = 9$) + HBPTL	3	WcMxG[t]D ($r = 5$) + HBPTL	5
WaMxA[t]D ($r = 9$) + SO1	3	WcSq[t]GD ($r = 13$) + SO1	5
BSq[t]GD ($r = 11$) + SO2	3	BSq[t]GD ($r = 7$) + HBPTL	5
WaSq[t]GD ($r = 11$) + SO2	3	WaSq[t]GD ($r = 9$) + HBPG	5
WaSq[t]GD ($r = 9$) + HBPTL	3	WcMxG[t]D ($r = 5$) + HBPG	5
BMxA[t]D ($r = 7$) + SO2	3	BSq[t]GD ($r = 7$) + HBPG	5
BMxG[t]D ($r = 7$) + SO1	3	WaMxG[t]D ($r = 7$) + HBPTL	5
WcSq[t]GD ($r = 11$) + SO1	3	WaSq[t]GD ($r = 9$) + SO1	5
WaSq[t]GD ($r = 11$) + HBPG	3	WcSq[t]GD ($r = 15$) + HBPG	5
WaMxA[t]D ($r = 7$) + HBPG	3	BS[t]AD ($r = 15$) + SO1	5
WaSq[t]GD ($r = 9$) + SO2	3	WaMxA[t]D ($r = 5$) + HBPG	5
WcSq[t]GD ($r = 9$) + SO2	3	WaMxA[t]D ($r = 7$) + SO1	5
WcSq[t]GD ($r = 11$) + SO2	3	BMxG[t]D ($r = 5$) + HBPG	5
BSq[t]GD ($r = 9$) + SO1	3	BS[t]AD ($r = 13$) + SO1	5
BMxG[t]D ($r = 7$) + SO2	3	BS[t]AD ($r = 15$) + SO2	5

Table 5.33: 300 top ranked algorithms on the data set with high magnitude noise, Table 2 of 3. Rankings calculated with cluster ranking at 95% confidence.

Algorithm	**Rank**	**Algorithm**	**Rank**
BS[t]AD ($r = 15$) + HBPTL	5	BS[t]AD ($r = 9$) + HBPTL	6
BS[t]AD ($r = 13$) + HBPTL	5	WcS[t]AD ($r = 15$) + SO2	6
WcMxA[t]D ($r = 3$) + SO2	5	WcS[t]AD ($r = 7$) + HBPTL	6
MMxA[t]D ($r = 3$) + HBPTL	5	WaMxG[t]D ($r = 5$) + HBPTL	6
WsMxA[t]D ($r = 5$) + SO2	5	BMxG[t]D ($r = 3$) + HBPTL	6
MMxA[t]D ($r = 5$) + SO2	5	WsMxG[t]D ($r = 5$) + HBPG	6
MMxA[t]D ($r = 3$) + SO2	5	WcS[t]AD ($r = 9$) + HBPG	6
BSq[t]GD ($r = 5$) + HBPG	6	WcS[t]AD ($r = 13$) + SO2	6
WaMxG[t]D ($r = 7$) + HBPG	6	BS[t]AD ($r = 9$) + SO2	6
WcMxA[t]D ($r = 5$) + SO1	6	WsMxG[t]D ($r = 3$) + HBPG	6
WaMxA[t]D ($r = 5$) + HBPTL	6	WcS[t]AD ($r = 11$) + HBPG	6
BS[t]AD ($r = 13$) + SO2	6	BS[t]AD ($r = 9$) + SO1	6
BS[t]AD ($r = 11$) + HBPTL	6	WcS[t]AD ($r = 13$) + HBPG	6
WcMxA[t]D ($r = 3$) + HBPG	6	WsMxA[t]D ($r = 9$) + HBPG	6
WcSq[t]GD ($r = 5$) + HBPG	6	WcSq[t]GD ($r = 3$) + HBPG	6
WsMxA[t]D ($r = 5$) + HBPG	6	WaS[t]AD ($r = 13$) + SO1	6
WcMxG[t]D ($r = 3$) + HBPG	6	WsMxG[t]D ($r = 3$) + HBPTL	6
WaSq[t]GD ($r = 7$) + HBPG	6	WcS[t]AD ($r = 15$) + HBPTL	6
WaSq[t]GD ($r = 7$) + SO2	6	WaS[t]AD ($r = 11$) + SO2	6
WcMxG[t]D ($r = 5$) + SO1	6	WsMxA[t]D ($r = 7$) + SO1	6
BS[t]AD ($r = 11$) + SO1	6	WcS[t]AD ($r = 11$) + SO1	6
BMxA[t]D ($r = 3$) + HBPG	6	BS[t]AD ($r = 9$) + HBPG	6
BS[t]AD ($r = 15$) + HBPG	6	WcS[t]AD ($r = 13$) + SO1	6
BS[t]AD ($r = 13$) + HBPG	6	WaMxA[t]D ($r = 3$) + HBPTL	6
BSq[t]GD ($r = 5$) + HBPTL	6	WaS[t]AD ($r = 11$) + HBPTL	6
WcS[t]AD ($r = 9$) + SO2	6	WsMxG[t]D ($r = 7$) + HBPG	6
BS[t]AD ($r = 11$) + SO2	6	WaMxA[t]D ($r = 5$) + SO2	6
WaS[t]AD ($r = 15$) + SO2	6	MMxA[t]D ($r = 7$) + HBPG	6
WcSq[t]GD ($r = 5$) + HBPTL	6	BMxG[t]D ($r = 3$) + HBPG	6
WaS[t]AD ($r = 15$) + HBPTL	6	MMxA[t]D ($r = 1$) + HBPG	6
BMxG[t]D ($r = 5$) + SO1	6	MMxG[t]D ($r = 5$) + HBPG	6
MMxA[t]D ($r = 5$) + HBPG	6	WcS[t]AD ($r = 7$) + HBPG	6
WsMxA[t]D ($r = 7$) + HBPG	6	WaSq[t]GD ($r = 5$) + SO2	6
WaS[t]AD ($r = 15$) + SO1	6	WaSq[t]GD ($r = 5$) + HBPTL	6
WcSq[t]GD ($r = 3$) + HBPTL	6	WaS[t]AD ($r = 9$) + HBPTL	6
WsMxA[t]D ($r = 3$) + HBPG	6	WcMxG[t]D ($r = 3$) + SO2	6
BS[t]AD ($r = 11$) + HBPG	6	BMxA[t]D ($r = 3$) + SO2	6
WaMxG[t]D ($r = 7$) + SO1	6	WsMxA[t]D ($r = 9$) + SO2	6
WcS[t]AD ($r = 11$) + HBPTL	6	WaMxG[t]D ($r = 5$) + SO2	6
WaS[t]AD ($r = 13$) + HBPTL	6	WaMxG[t]D ($r = 5$) + HBPG	6
WcS[t]AD ($r = 11$) + SO2	6	WcS[t]AD ($r = 15$) + SO1	6
WaS[t]AD ($r = 13$) + SO2	6	WsMxA[t]D ($r = 5$) + SO1	6
MMxA[t]D ($r = 3$) + HBPG	6	MMxG[t]D ($r = 5$) + SO2	6
WcSq[t]GD ($r = 5$) + SO1	6	WcS[t]AD ($r = 9$) + SO1	6
WcS[t]AD ($r = 13$) + HBPTL	6	MMxA[t]D ($r = 7$) + SO2	6
WcS[t]AD ($r = 9$) + HBPTL	6	WsMxA[t]D ($r = 3$) + SO2	6
BSq[t]GD ($r = 5$) + SO1	6	WcSq[t]GD ($r = 3$) + SO2	6
MMxG[t]D ($r = 3$) + HBPG	6	WcS[t]AD ($r = 15$) + HBPG	7
BS[t]AD ($r = 7$) + HBPTL	6	BSq[t]GD ($r = 3$) + HBPTL	7
WaSq[t]GD ($r = 7$) + SO1	6	MMxA[t]D ($r = 5$) + HBPTL	7

Table 5.34: 300 top ranked algorithms on the data set with high magnitude noise, Table 3 of 3. Rankings calculated with cluster ranking at 95% confidence.

- When creating a truncated match cost function for use in an environment with high magnitude sensor noise it is often more beneficial to truncate the result at the colour channel level (form XY[t]Z) than to truncate the result after the per-channel results have been combined (form X[t]YZ);
- It is almost always better to calculate the difference between two greyscale intensity levels by using a difference operator rather than the Birchfield and Tomasi measure when the input contains at most "low" magnitude sensor noise;
- The generalized exponential should always be used instead of an L1 or L2 norm due to the flexibility gained from the parametrization of the exponent;
- When combining greyscale match cost functions into a colour match cost function for use on data containing at most "low" magnitude sensor noise a simple max or square root of summation is usually sufficient;
- When combining greyscale match cost functions into a colour match cost function for use on data containing high magnitude noise the flexibility of a weighted sum will result in a superiour match cost function if appropriate weights can be found;
- In RGB colour space, the weights of this weighted sum should be chosen such that they assign a low weight to the blue colour channel;
- Applying any one of the five spatial aggregates that we consider in this study to a match cost function is likely to improve the accuracy of the global stereopsis algorithm that is being used;
- The *bilateral* spatial aggregate is most likely to yield the best accuracy improvement amongst the five spatial aggregates we consider in this study.

Our analysis indicates an apparent relationship between the general shape of a greyscale match cost function and the method used to combine the results from that match cost function to yield a colour match cost function. Furthermore, an analysis of the relative performance of the five spatial aggregates used in this study provide some insight into the relative effectiveness of the spatial and spectral weighting terms used definitions of these spatial aggregates. Our analysis indicates that both weighting functions contribute to a high quality spatial aggregate, but in different ways. The spatial weight term seems to act as a hedge against choosing too large of a spatial radius over which to aggregate a match cost function; that is, it limits the negative effects that can result from including match cost information that is too far from the pixel at the center of the aggregation window. The spectral weight term acts as a rudimentary object detector by assigning a higher weight to those pixels that have the same/similar colour. Our analysis indicates that this weighting term is highly useful in creating a high quality spatial aggregate.

Finally, we provide rankings of all of the stereopsis algorithms used in this study and identify those algorithms that consistently yield the highest accuracy disparity maps as the magnitude of noise in the input is increased.

Chapter 6

The Effects of Colour Representation on Global Stereopsis

The second goal in this dissertation is to seek answers to the questions:

- How much can the accuracy of a global stereopsis algorithm be improved by only changing the colour representation of the input images?
- Which colour representation offers the highest degree of accuracy improvement when used in this manner?

When designing algorithms for stereopsis the vast majority of researchers use input images with colour represented in the RGB colour space. However, alternative colour representations have been shown to improve the accuracy of many computer vision algorithms, including stereopsis algorithms (see Section 3.3 for some examples). Knowing whether we can improve stereopsis algorithm accuracy by changing nothing about the algorithm itself, but, instead, changing the the colour representation used for the input images would be beneficial to the stereopsis research community as a whole. Such a change to an algorithm is so trivial to make that, should it improve algorithm accuracy, there is no reason that researchers should not incorporate it into their algorithm design.

To study these questions we use the same methodology that we employ to study match cost functions within global stereopsis algorithms. We use a small set of 15 match cost functions along with the five global stereopsis frameworks used in the match cost function study. The resulting 75 global stereopsis algorithms are then run using input in various colour representations, and the data generated is analyzed.

The data for this study was collected by running our program on the Glacier [99], Snowpatch [100], and Terminus [101] clusters of the Western Canada Research Grid. All three of these clusters run a version of the Linux operating system and are configured as Beowulf clusters. The Glacier cluster consists of 840 IBM eServer BladeCenter HS20 computers with dual 3.0GHz Xeon processors. The Snowpatch cluster consists of 16 HP BL460c Blade computers with two Intel Xeon X5355 2.66GHz quad-core processors each. The Terminus cluster is comprised of 468 BL465c G1 CTO Blade computers with dual-core 2.4 GHz AMD Opteron processors. This data collection required approximately 5.738 years of CPU time from the processors in these clusters.

We begin this Chapter with a description of our experimental design, and then move on to discuss results.

6.1 Experimental Design

For our study on the effect of input colour representation on global stereopsis algorithm accuracy we use 75 stereopsis algorithms derived from a combination of 15 match cost functions and the five global stereopsis frameworks described in Section 5.2. User-parameters for these stereopsis algorithms are optimized using the same response-surface based optimization method we employ in our match cost function study. The data set used for this comparison is our 57 image pair data set with no synthetic sensor noise introduced. We compare 30 colour representations in total.

6.1.1 Match Cost Functions Used

The 15 match cost functions we use are constructed from all permutations of the *max* (Mx), *sum* (S), *square root of sum* (Sq), *weighted sum* (Sw), and *sum minus max* (Sx) channel aggregates with the *generalized exponential* (G), *L1 norm* (A), and *L2 norm* (S) channel norm functions combined with the *difference* (D) channel function. The resulting functions are: MxAD, MxSD, MxGD, SAD, SSD, SGD, SqAD, SqSD, SqGD, SwAD, SwSD, SwGD, SxAD, SxSD, and SxGD. We do not include any truncated match cost functions in this study because we do not wish to risk tainting the results of our analysis by inadvertently choosing an inappropriate range of possible truncation values to use in the parameter optimization process.

6.1.2 Colour Representations Investigated

The colour representations we consider include (see Section 3.2 for a description of each): greyscale, RGB, CIEXYZ, CIExyY, CIELab, CIELuv, YCrCb, Ohta, HSV, HSVXYZ, HLS, HLSXYZ, $c_1c_2c_3$, $l_1l_2l_3$, $h_1h_2h_3$, Brockelbank, and Campbell. Since the magnitude of the hue channel value in the HSV and HLS colour representations is significantly higher than the other channel values in these representations we also include what we call the *HSV 1* and *HLS 1* colour representations. These two colour representations are derived from the HSV and HLS representations by scaling the H component of the colour to the range $[0, 1]$.

We include three colour representations based on applying the Karhunen-Loève (KL) transform to the RGB colour space: *KL Left*, *KL Right*, and *KL Joint*. Since the KL transform is derived from an image we want to determine whether it makes a difference if the transform is derived from just the left image, the right image, or both images in an input image pair. The KL Left, KL Right, and KL Joint colour representations are derived by calculating the KL transform of the RGB space representations of just the left, just the right, or both images in an image pair, respectively.

We also include colour representations that are derived by applying a KL transform to the CIELab and CIELuv colour spaces: LuvKL, and LabKL. The LuvKL colour space is derived by transforming the input images to CIELuv colour space, calculating the KL transform from both input images, and then applying the resulting transform. The LabKL colour space is derived in the same manner except that it transforms the input images to the CIELab colour space.

The final six colour spaces are: rRGB, rLab, rLuv, r_2RGB, r_2Lab, and r_2Luv. These colour spaces are derived by rotating either the RGB, CIELab, or CIELuv colour spaces. The rotations used are derived using one of the two methods described below.

When changing the colour representation used in the input images to our stereopsis algorithms we made no changes to the algorithms themselves. The exception to this is when converting the input images to the HSV, HLS, HSV 1, or HLS 1 colour representations. When the input is in one of these colour representations we alter the different channel function to identify when the difference is being taken of the H channel of the colour, and use our modified difference function for hues as described in Equation 3.10c of Section 3.2.4.

Furthermore, when the smoothing prior of the stereopsis framework calls for the distance between colours of one of these colour representations we use Equation 3.10c for calculating the difference between the two H values of the colours present in the equation for the Euclidean distance between two three dimensional points.

The rRGB, rLuv, and rLab Colour Representations

We create the rRGB, rLuv, and rLab colour representations by applying a rotation to the RGB, CIELuv, and CIELab colour spaces, respectively. The derivation for this rotation is inspired by the work of Ohta [71] in that he derives a linear transform of the RGB colour space from a set of Karhunen-Loève transforms. The idea being to derive colour representations that are more orthogonal, in an information theoretic sense [42], than the RGB, CIELuv, and CIELab spaces.

We begin the derivation of the rotation by calculating the covariance matrices for the 200 training images in the Berkeley segmentation database [66]; see Equation 3.19 in Section 3.2.10 for how to calculate the covariance matrix of an image. The Berkeley database is used due to its size and because it does not contain any images that we include in our own data set. The eigenvectors of these 200 covariance matrices are divided into three sets, E_1, E_2, and E_3, containing the dominant, secondary, and tertiary eigenvectors, respectively. From each of these sets, E_i, we derive vectors $\vec{\mathbf{v^i}}$:

1. Perform a mean-shift [23] on every vector, $\vec{\mathbf{v}} \in E_i$, with a spatial bandwidth of 0.075 and a Gaussian kernel to find the modes of the set.

2. $\vec{\mathbf{v^i}}$ is the mean of the modes of E_i.

We then apply the Gramm-Schmidt orthonormalization process to the set of vectors $\{\vec{\mathbf{v^1}}, \vec{\mathbf{v^2}}, \vec{\mathbf{v^3}}\}$ to obtain an orthonormal basis $\{\vec{\mathbf{u^1}}, \vec{\mathbf{u^2}}, \vec{\mathbf{u^3}}\}$:

$$\vec{\mathbf{u^1}} = \frac{\vec{\mathbf{v^1}}}{||\vec{\mathbf{v^1}}||} \tag{6.1a}$$

$$\vec{\mathbf{u^{2\prime}}} = \vec{\mathbf{v^2}} - (\vec{\mathbf{v^2}} \cdot \vec{\mathbf{u^1}})\vec{\mathbf{u^1}} \tag{6.1b}$$

$$\vec{\mathbf{u^2}} = \frac{\vec{\mathbf{u^{2\prime}}}}{||\vec{\mathbf{u^{2\prime}}}||} \tag{6.1c}$$

$$\vec{\mathbf{u^{3\prime}}} = \vec{\mathbf{v^3}} - (\vec{\mathbf{v^3}} \cdot \vec{\mathbf{u^2}})\vec{\mathbf{u^2}} - (\vec{\mathbf{v^3}} \cdot \vec{\mathbf{u^1}})\vec{\mathbf{u^1}} \tag{6.1d}$$

$$\vec{\mathbf{u^3}} = \frac{\vec{\mathbf{u^{3\prime}}}}{||\vec{\mathbf{u^{3\prime}}}||}. \tag{6.1e}$$

The rotation is the change of basis transform to the orthonormal basis $\{\vec{\mathbf{u^1}}, \vec{\mathbf{u^2}}, \vec{\mathbf{u^3}}\}$:

$$\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} \vec{\mathbf{u^1}}_1 & \vec{\mathbf{u^1}}_2 & \vec{\mathbf{u^1}}_3 \\ \vec{\mathbf{u^2}}_1 & \vec{\mathbf{u^2}}_2 & \vec{\mathbf{u^2}}_3 \\ \vec{\mathbf{u^3}}_1 & \vec{\mathbf{u^3}}_2 & \vec{\mathbf{u^3}}_3 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix}. \tag{6.2}$$

Applying this process to the Berkeley database in the RGB colour space, we obtain the rotation to obtain a colour, (r', g', b'), for the rRGB colour space from a colour, (r, g, b), in RGB colour space:

$$\begin{pmatrix} r' \\ g' \\ b' \end{pmatrix} = \begin{pmatrix} -0.540071 & 0.798641 & -0.265512 \\ -0.579616 & -0.124210 & 0.805368 \\ 0.610220 & 0.588851 & 0.529987 \end{pmatrix} \begin{pmatrix} r \\ g \\ b \end{pmatrix}. \tag{6.3}$$

Applying this process to the Berkeley database in the CIELuv colour space, we obtain the rotation to obtain a colour, (L', u', v'), for the rLuv colour space from a colour, (L, u, v), in CIELuv colour space:

$$\begin{pmatrix} L' \\ u' \\ v' \end{pmatrix} = \begin{pmatrix} 0.995939 & 0.0582081 & 0.0686899 \\ 0.0882167 & -0.783456 & -0.615154 \\ 0.0180086 & 0.618715 & -0.785409 \end{pmatrix} \begin{pmatrix} L \\ u \\ v \end{pmatrix}. \tag{6.4}$$

Finally, applying this process to the Berkeley database in the CIELab colour space, we obtain the rotation to obtain a colour, (L', a', b'), for the rLab colour space from a colour, (L, a, b), in CIELab colour space:

$$\begin{pmatrix} L' \\ a' \\ b' \end{pmatrix} = \begin{pmatrix} 0.994981 & 0.0491930 & -0.0871347 \\ 0.0910086 & -0.806856 & 0.583696 \\ -0.0415914 & -0.588696 & -0.807284 \end{pmatrix} \begin{pmatrix} L \\ a \\ b \end{pmatrix}. \tag{6.5}$$

The r$_2$RGB, r$_2$Luv, and r$_2$Lab Colour Representations

The r$_2$RGB, r$_2$Luv, and r$_2$Lab colour representations are also created by applying a rotation to the RGB, CIELuv, and CIELab colour spaces, respectively. The derivation of this rotation is identical to the derivation of the rotation for the rRGB, rLuv, and rLab colour representations except in the way that the vectors $\vec{\mathbf{v}}^{\mathbf{i}}$ are derived from the sets of eigenvectors, E_i.

For the r$_2$RGB, r$_2$Luv, and r$_2$Lab colour representations we derive the vectors $\vec{\mathbf{v}}^{\mathbf{i}}$ from the sets E_i by:

1. Perform a mean-shift [23] on every vector, $\vec{\mathbf{v}} \in E_i$, with a spatial bandwidth of 0.075 and a Gaussian kernel to find the modes of the set.

2. Apply Parzen-window [72] kernel density estimation with a Gaussian kernel and a spatial bandwidth of 0.075 to each of these modes to obtain the values of the probability density function of the vectors in E_i at each mode.

3. $\vec{\mathbf{v}}^{\mathbf{i}}$ is the mode with highest probability.

Applying this process to the Berkeley database in the RGB colour space, we obtain the rotation to obtain a colour, (r', g', b'), for the r$_2$RGB colour space from a colour, (r, g, b), in RGB colour space:

$$\begin{pmatrix} r' \\ g' \\ b' \end{pmatrix} = \begin{pmatrix} -0.418234 & 0.805652 & -0.419529 \\ -0.668514 & 0.0396742 & 0.742640 \\ 0.614954 & 0.591059 & 0.521997 \end{pmatrix} \begin{pmatrix} r \\ g \\ b \end{pmatrix}. \tag{6.6}$$

Applying this process to the Berkeley database in the CIELuv colour space, we obtain the rotation to obtain a colour, (L', u', v'), for the r$_2$Luv colour space from a colour, (L, u, v), in CIELuv colour space:

$$\begin{pmatrix} L' \\ u' \\ v' \end{pmatrix} = \begin{pmatrix} 0.995356 & 0.0800828 & 0.0534183 \\ 0.0944942 & -0.706902 & -0.700971 \\ -0.0183742 & 0.702763 & -0.711186 \end{pmatrix} \begin{pmatrix} L \\ u \\ v \end{pmatrix}. \tag{6.7}$$

Finally, applying this process to the Berkeley database in the CIELab colour space, we obtain the rotation to obtain a colour, (L', a', b'), for the r$_2$Lab colour space from a colour, (L, a, b), in CIELab colour space:

$$\begin{pmatrix} L' \\ a' \\ b' \end{pmatrix} = \begin{pmatrix} 0.998972 & 0.0440508 & -0.0107370 \\ -0.00613855 & -0.103228 & -0.994639 \\ 0.0449230 & -0.993682 & 0.102851 \end{pmatrix} \begin{pmatrix} L \\ a \\ b \end{pmatrix}. \tag{6.8}$$

6.2 Analysis

The goals in our analysis for this study are to:

1. Determine which colour spaces can be expected to yield an improvement in global stereopsis algorithm accuracy over the RGB colour space;
2. Identify some of the properties of a global algorithm that can act as indicators for when a colour space can be expected to yield an improvement in accuracy over the RGB colour space; and
3. To determine how much of an improvement can be expected.

We evaluate the accuracy of each disparity map generated in this study in the same manner used in our study on match cost functions for global stereopsis algorithms. That is, we calculate the percentage of pixels that differ by more than one from the given ground truth disparity over a mask image (hereafter referred to as the *error rate*); see Equation 4.2. The mask image that we use for this evaluation contains zero-valued pixels only where the ground truth disparity map has no valid disparity assigned and in a small border on either the left or right side of the image. This border is equal in width to the baseline separation between the two images in the image pair, is on the left side in the left image, and the right side in the right image.

Since this analysis is focused on determining which colour spaces yield an improvement in accuracy over the results obtained when using the RGB colour space we consider the percentage accuracy improvement over the results obtained from the RGB colour space. Given that one of our calculated disparity maps, f, has an error rate of ε and the disparity map for the same image as calculated by the same algorithm in the RGB colour space has error rate ε_{rgb} then we calculate the percent accuracy improvement for f as:

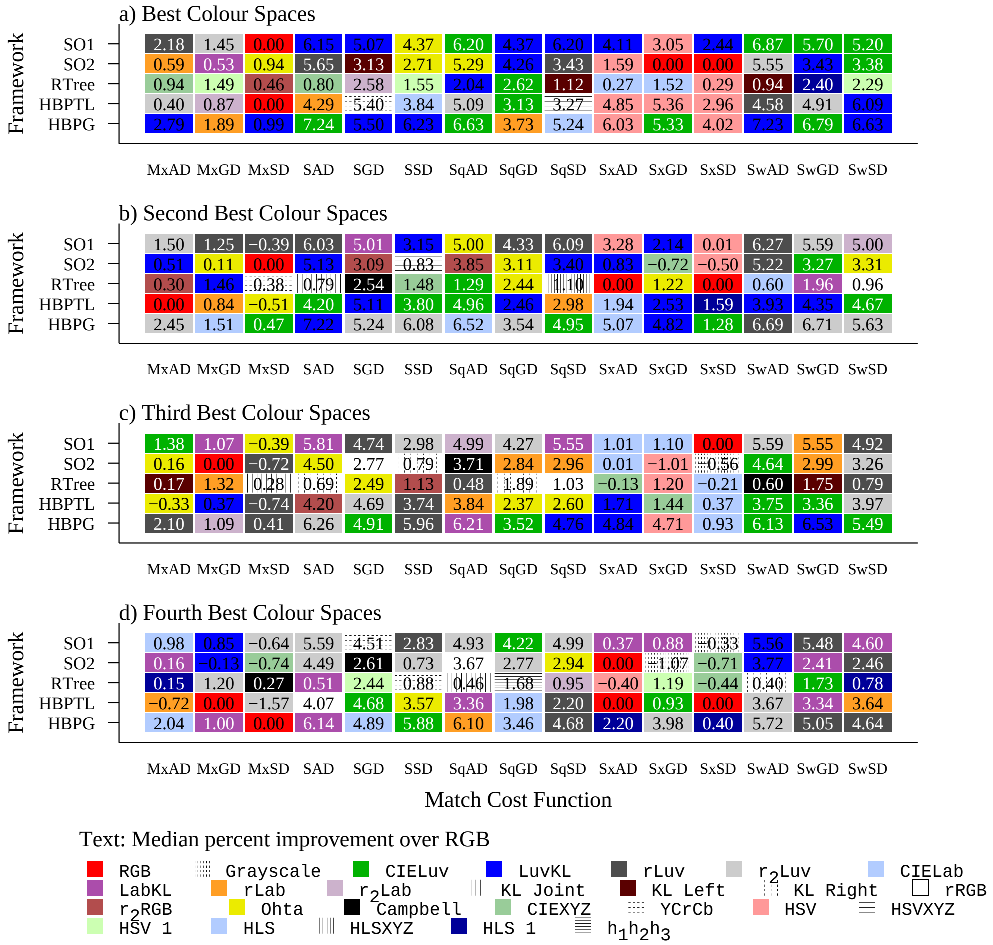
$$\text{Percent accuracy improvement} = 100\frac{\varepsilon_{\text{rgb}} - \varepsilon}{\varepsilon_{\text{rgb}}}. \tag{6.9}$$

Calculated this way, a negative percent accuracy improvement indicates a decrease in accuracy over the RGB colour space.

In this section we divide our analysis into four parts. In the first part we discuss some of the accuracy improvement results for individual global stereopsis algorithms. In the second and third parts we discuss the aggregate improvement of each colour space by match cost function, and by global stereopsis framework, respectively. We conclude this section with a discussion of the over all improvements observed in this study.

6.2.1 Analysis by Algorithm

Figure 6.1 shows the four best colour spaces, by median accuracy improvement, for each of the 75 global stereopsis algorithms in this study and their median accuracy improvement in those colour spaces; the median accuracy improvements in this diagram are taken over all 114 (57 image pairs times two images per image pair) images in our data set. Immediately obvious in this figure is that a large number of algorithms can gain substantial improvements in their accuracy by simply changing the colour space of their input to either the CIELuv (dark green with white text in the figure) or LuvKL (blue with black text in the figure) colour spaces. A lesser number of algorithms also gain substantial median accuracy improvements in the r_2Luv (light grey with black text), rLuv (dark grey with white text), LabKL (purple with white text), CIELab (cyan with black text), rLab (orange with black text), r_2Lab (light purple with black text), Ohta (yellow with black text), and HSV (pink with black text) colour spaces.



a) Best Colour Spaces

Framework	MxAD	MxGD	MxSD	SAD	SGD	SSD	SqAD	SqGD	SqSD	SxAD	SxGD	SxSD	SwAD	SwGD	SwSD
SO1	2.18	1.45	0.00	6.15	5.07	4.37	6.20	4.37	6.20	4.11	3.05	2.44	6.87	5.70	5.20
SO2	0.59	0.53	0.94	5.65	3.13	2.71	5.29	4.26	3.43	1.59	0.00	0.00	5.55	3.43	3.38
RTree	0.94	1.49	0.46	0.80	2.58	1.55	2.04	2.62	1.12	0.27	1.52	0.29	0.94	2.40	2.29
HBPTL	0.40	0.87	0.00	4.29	5.40	3.84	5.09	3.13	3.27	4.85	5.36	2.96	4.58	4.91	6.09
HBPG	2.79	1.89	0.99	7.24	5.50	6.23	6.63	3.73	5.24	6.03	5.33	4.02	7.23	6.79	6.63

b) Second Best Colour Spaces

Framework	MxAD	MxGD	MxSD	SAD	SGD	SSD	SqAD	SqGD	SqSD	SxAD	SxGD	SxSD	SwAD	SwGD	SwSD
SO1	1.50	1.25	−0.39	6.03	5.01	3.15	5.00	4.33	6.09	3.28	2.14	0.01	6.27	5.59	5.00
SO2	0.51	0.11	0.00	5.13	3.09	0.83	3.85	3.11	3.40	0.83	−0.72	−0.50	5.22	3.27	3.31
RTree	0.30	1.46	0.38	0.79	2.54	1.48	1.29	2.44	1.10	0.00	1.22	0.00	0.60	1.96	0.96
HBPTL	0.00	0.84	−0.51	4.20	5.11	3.80	4.96	2.46	2.98	1.94	2.53	1.59	3.93	4.35	4.67
HBPG	2.45	1.51	0.47	7.22	5.24	6.08	6.52	3.54	4.95	5.07	4.82	1.28	6.69	6.71	5.63

c) Third Best Colour Spaces

Framework	MxAD	MxGD	MxSD	SAD	SGD	SSD	SqAD	SqGD	SqSD	SxAD	SxGD	SxSD	SwAD	SwGD	SwSD
SO1	1.38	1.07	−0.39	5.81	4.74	2.98	4.99	4.27	5.55	1.01	1.10	0.00	5.59	5.55	4.92
SO2	0.16	0.00	−0.72	4.50	2.77	0.79	3.71	2.84	2.96	0.01	−1.01	−0.56	4.64	2.99	3.26
RTree	0.17	1.32	0.28	0.69	2.49	1.13	0.48	1.89	1.03	−0.13	1.20	−0.21	0.60	1.75	0.79
HBPTL	−0.33	0.37	−0.74	4.20	4.69	3.74	3.84	2.37	2.60	1.71	1.44	0.37	3.75	3.36	3.97
HBPG	2.10	1.09	0.41	6.26	4.91	5.96	6.21	3.52	4.76	4.84	4.71	0.93	6.13	6.53	5.49

d) Fourth Best Colour Spaces

Framework	MxAD	MxGD	MxSD	SAD	SGD	SSD	SqAD	SqGD	SqSD	SxAD	SxGD	SxSD	SwAD	SwGD	SwSD
SO1	0.98	0.85	−0.64	5.59	4.51	2.83	4.93	4.22	4.99	0.37	0.88	−0.33	5.56	5.48	4.60
SO2	0.16	−0.13	−0.74	4.49	2.61	0.73	3.67	2.77	2.94	0.00	−1.07	−0.71	3.77	2.41	2.46
RTree	0.15	1.20	0.27	0.51	2.44	0.88	0.46	1.68	0.95	−0.40	1.19	−0.44	0.40	1.73	0.78
HBPTL	−0.72	0.00	−1.57	4.07	4.68	3.57	3.36	1.98	2.20	0.00	0.93	0.00	3.67	3.34	3.64
HBPG	2.04	1.00	0.00	6.14	4.89	5.88	6.10	3.46	4.68	2.20	3.98	0.40	5.72	5.05	4.64

Figure 6.1: The four best colour spaces, by median percent accuracy improvement, for each global stereopsis algorithm.

This figure illustrates, quite clearly, that the colour space that yields the best accuracy gains is dependent upon the specific global stereopsis algorithm being used. However, there are also observable patterns in which colour spaces are beneficial to different stereopsis frameworks and different match cost functions. We discuss these patterns in the next sections.

For additional information, Appendix K contains box plots of the distribution of percent accuracy improvements over RGB over the entire data set for each algorithm in each colour space. The tails of the plots indicate the fifth and 95^{th} percentiles, the box extents indicate the 25^{th} and 75^{th} percentiles, and the line in the box indicates the location of the median of the distribution. We also include the median value and the percentage of the data points that are greater than zero on the right side of each plot.

6.2.2 Analysis by Stereopsis Framework

As discussed in chapter 5, algorithms based on the RTree framework derive most of their accuracy from the quality of the segmentation used. Our application of colour spaces to algorithms in this framework does not change the colour space used by the segmentation procedure. In light of this, the data in Figures 6.1 and I.3 (see Appendix I), and baring a high level of synergy between the match cost function and a colour space, we do not expect a change of input colour space to benefit these algorithms; on the other hand, it would likely be beneficial to investigate which colour space yields the most beneficial segmentations when the input to the segmentation component is changed to use that colour space.

In aggregate, algorithms based on the SO1 global stereopsis frameworks exhibit median improvements in accuracy of over 2.5% in the CIELuv, LuvKL, rLuv, r_2Luv, and LabKL colour spaces (see Figure I.1). These algorithms also exhibit lesser, but positive, median accuracy improvements in the CIELab, rLab, r_2Lab, and Ohta colour spaces. Similarly, the algorithms based on the HBPG framework (see Figure I.5) exhibit a median accuracy improvement of 2.7% to 4.1% in all of these colour spaces, except the Ohta space.

The SO1 and HBPG global stereopsis frameworks both utilize a smoothing term that determines its value by applying a threshold to the magnitude of the colour gradient between two neighbouring pixels. Thus, our data suggests that gradient-threshold smoothing terms, such as those in these two frameworks, are likely to benefit from perceptually uniform colour spaces such as CIELab and CIELuv.

The smoothing term in the SO2 framework uses gradient magnitude information in a different manner. In this framework, the smoothing term is partially modulated by the gradient magnitude normalized by the maximum gradient magnitude over the input image. In aggregate, the algorithms based on this framework exhibit some median gains in accuracy of more than 1.2% in the LuvKL, rLuv, and Ohta colour spaces. They also exhibit some lesser gains in the r_2Luv, LabKL, and rLab colour spaces. In any case, the gains exhibited in these algorithms are not as substantial as the gains in the algorithms utilizing the SO1 and HBPG frameworks; suggesting that the smoothing term in the SO1 framework does not have much to gain from a change in colour space.

The smoothing term in the HBPTL framework does not utilize gradient information in any way. Instead, the smoothing term in this framework must act in synergy with the match cost function to enforce smoothness. In aggregate, the algorithms based on the HBPTL framework exhibit (see Figure I.4) only marginal accuracy improvement by changing the colour space of the input to something other than RGB.

6.2.3 Analysis by Match Cost Function

Consulting Figure 6.1 we see that there are definite patterns in which colour spaces are a benefit to which match cost functions. Furthermore, this pattern appears to be dependent on which channel aggregate is used in the match cost function.

Algorithms that use a match cost function based on the max (Mx) channel aggregate do not appear to exhibit any meaningful improvement in accuracy when changing the input colour space to something other than RGB. By using a max channel aggregate, these match cost functions base their match cost on the largest channel norm value taken over the three colour channels in a pixel's colour. Thus, our data indicates that as long as the elements of a colour vector contain information that is suitable for measuring colour difference, there is not much difference in which colour representation is used with match cost functions that use a max channel aggregate.

Algorithms that use a match cost function based on the sum (S), square root of sum (Sq), and weighted sum (Sw) channel aggregates all derive substantial improvements in accuracy by utilizing a perceptually uniform colour space instead of RGB. Note that perceptual uniformity is an invariant under rotation, and that the Karhunen-Loève transform is simply a rotation. Thus, the LuvKL, LabKL, rLuv, rLab, r_2Luv, and r_2Lab colour spaces are also perceptually uniform. All of these match cost functions can be considered to be similar in style to the L1 and Euclidean distance metrics (in fact, SqSD is the Euclidean distance), so it is not entirely surprising that it would be beneficial to use colour spaces where Euclidean distance is a close approximation to our perception of colour difference.

Interestingly, algorithms that use a match cost function based on the sum minus max (Sx) channel aggregate can generally have their accuracy improved by using the HSV colour space for the input images. These are the only match cost functions where this is the case. Recall that the HSV colour space represents colour as the triple (H, S, V) where the H (hue) component is an angle in the range $[0, 360)$, and the S (saturation) and V (value or brightness/luminance) components are in the range $[0, 1]$. This significant difference in the ranges of the components of an HSV colour mean that any time the difference between two colours in HSV space of even only slightly different hue is taken, the difference in the H component will undoubtedly be the highest of the three component differences. The improvements observed in our data by using the HSV colour space with match cost functions utilizing the Sx channel aggregate suggest that there may be sufficient information in only the saturation and brightness of pair of colours to sufficiently distinguish them for stereopsis; further investigation is required.

For the interested reader, further information on the aggregate relative performance of colour spaces for each match cost function can be found in the box plots of Appendix J.

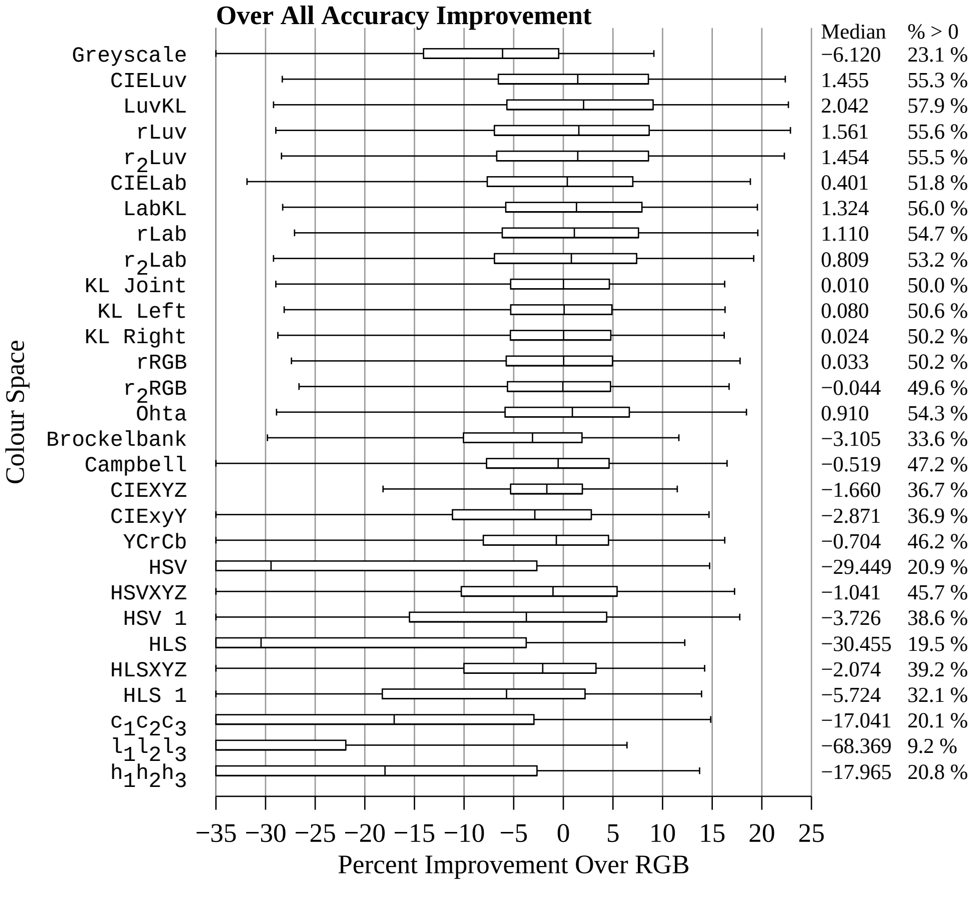


Figure 6.2: Box plots of the over all aggregate percent accuracy improvement distributions for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

6.2.4 Over All Analysis

In Figure 6.2 we present box plots depicting the distribution of percent accuracy improvements, over the RGB colour space, over all 75 global stereopsis and all image pairs in our data set for each colour space. The data in this figure gives us a sense of which colour spaces can be expected to yield an improvement, or drop, in accuracy void of any details of the individual global stereopsis algorithm being considered. This data indicates that nine of the 29 colour spaces that we compare in this study can be expected to yield some sort of improvement in accuracy over the RGB colour space when the input images are converted to them: CIELuv, LuvKL, rLuv, $\mathrm{r_2}$Luv, CIELab, LabKL, rLab, $\mathrm{r_2}$Lab, and Ohta. The data also indicates five colour spaces are likely unfit for use in stereo without specially tailored distance formula: HSV, HLS, $c_1c_2c_3$, $l_1l_2l_3$, and $h_1h_2h_3$.

In general, our data indicates that perceptual uniformity is a beneficial colour space property for global stereopsis algorithms. Furthermore, recall that the Ohta colour space was developed from observations of a series of Karhunen-Loève (KL) transforms calculated during application of a segmentation algorithm to a small set of images. The accuracy gains observed by changing the input images to the Ohta colour space and colour spaces that apply a KL transform to the CIELuv and CIELab colour spaces (LuvKL and LabKL, respectively) suggest that there may also be a benefit to utilizing more orthogonal, in an information theoretic sense, colour spaces in global stereopsis algorithms.

Additionally, our own colour spaces that rotate the CIELuv, CIELab, and RGB colour spaces (the r- and $\mathrm{r_2}$- colour spaces) exhibit similar accuracy improvements to their non-rotated base spaces. This suggests that our approach to calculating a generalized rotation of the CIELab, CIELuv, and RGB spaces is unlikely to be of benefit to global stereopsis algorithms.

Finally, our reason for including the KL Left, KL Right, and KL Joint colour spaces is to determine whether or not it makes a difference if the KL transform is derived from just the left, just the right, or both images in an image pair. Our data indicates that there is no significant difference between these three options.

6.3 Summary

In this chapter we present the results from, and analysis for, our study on the effects of colour representation on global stereopsis. In this study, we consider the accuracy of 75 global stereopsis algorithms on a data set of 57 image pairs in 30 colour spaces. The goal in this study is to determine whether simply changing the colour representation used in the input to a global stereopsis algorithm can yield an improvement in the algorithm's accuracy, and, if so, by approximately how much.

Our analysis indicates that in most cases an improvement in a global stereopsis algorithm's accuracy can be expected by changing the colour representation of the input images to a perceptually uniform colour space such as CIELuv or CIELab. Furthermore, we observe median improvements in accuracy of as much as 7.2% by changing the input images to a perceptually uniform colour space for some algorithms.

In general, our analysis shows that global stereopsis algorithms that incorporate a smoothing term that calculates its value by applying a threshold to the gradient magnitude in the colour between neighbouring pixels, and those that incorporate a match cost function similar in style to the L1 or Euclidean distance metrics stand to gain the most by simply switching the colour space of the input to a perceptually uniform colour space.

We also show that there are some colour spaces that are generally unsuitable for being used in a direct substitution such as we perform in this study. These colour spaces are the HSV, HLS, $c_1c_2c_3$, $l_1l_2l_3$, and $h_1h_2h_3$ colour spaces. However, all is not lost. Our data indicates that there may be sufficient information in the saturation and value components of the HSV colour space to sufficiently distinguish colours for the purpose of global stereopsis. Furthermore, using a distance metric that is specific to these colour spaces may still prove to be fruitful. Both of these avenues of investigation are worthwhile future research.

Chapter 7

Conclusion and Future Work

Our goal in this dissertation has been to further our understanding of the components of global stereopsis algorithms. To this end we conduct two, first of their kind, large scale studies on two of these components: match cost functions, and the colour space used to represent colour in the input images.

For these studies we develop a novel evaluation framework for stereopsis algorithms to evaluate the relative performance of stereopsis algorithms. To ensure that the evaluation is being performed on a level playing field, our framework incorporates a response-surface method for optimizing user-parameters in the stereopsis algorithms being compared. Response-surface methods iteratively build a model of an underlying objective function that is used to determine the next point to evaluate the objective function at for inclusion into the model. Large scale studies such as the ones conducted in this dissertation require such automated methods of parameter optimization due to input-sensitivity of some user-parameters. As part of this evaluation framework we also propose a ranking algorithm, that we call *cluster ranking*, that will assign the same rank to algorithms that produce results that cannot be statistically differentiated.

The first study we conduct is a systematic comparison of the relative performance of several hundred match cost functions, in the context of five different global stereopsis frameworks, on a large data set. This data set consists of a combination of synthetic (ray traced) and photographic image pairs with three different variance-magnitudes of synthetic noise introduced: none, “low” magnitude, and “high” magnitude. The match cost functions that we use in this study are built from component parts in such a way that we are able to make observations about the relative performance of those parts by analyzing the relative performance of the match cost functions that use them.

Our analysis of the data from this study indicates that there is likely no “Swiss army knife” match cost function that will be ideal for every situation. Instead, the match cost function to use in a global stereopsis algorithm must be chosen based on the properties of the algorithm to ensure the best possible accuracy from the resulting algorithm. That said, we have identified some general guidelines for choosing a decent match cost function for global stereopsis algorithms. Specifically:

- The choice of match cost function used in a segmentation based formulation of stereopsis is likely to be irrelevant;
- Making a match cost function more robust to sensor noise and outliers by truncating it, in some manner, will almost always result in a better match cost function;
- When creating a truncated match cost function for use in an environment with high magnitude sensor noise it is often more beneficial to truncate the result at the colour

channel level (form XY[t]Z) than to truncate the result after the per-channel results have been combined (form X[t]YZ);

- It is almost always better to calculate the difference between two greyscale intensity levels by using a difference operator rather than the Birchfield and Tomasi measure when the input contains at most "low" magnitude sensor noise;
- The generalized exponential should always be used instead of an L1 or L2 norm due to the flexibility gained from the parametrization of the exponent;
- When combining greyscale match cost functions into a colour match cost function for use on data containing at most "low" magnitude sensor noise a simple max or square root of summation is usually sufficient;
- When combining greyscale match cost functions into a colour match cost function for use on data containing high magnitude noise the flexibility of a weighted sum will result in a superiour match cost function if appropriate weights can be found;
- In RGB colour space, the weights of this weighted sum should be chosen such that they assign a low weight to the blue colour channel;
- Applying any one of the five spatial aggregates that we consider in this study to a match cost function is likely to improve the accuracy of the global stereopsis algorithm that is being used;
- The *bilateral* spatial aggregate is most likely to yield the best accuracy improvement amongst the five spatial aggregates we consider in this study.

We also identify an apparent relationship between the general shape of a greyscale match cost function's response curve and the method used to combine the results from that match cost function into a colour match cost function. By comparing the relative performance of five spatial aggregates for match cost functions we are able to gain some insight into the relative effectiveness of two weighting terms that are used in the definitions of these aggregates: a Gaussian spatial weighting term, and Gaussian spectral weighting term. Our analysis indicates that both of these weighting terms contribute to high quality spatial aggregates, but in different ways. The spatial weight term seems to act as a hedge against choosing too large of a spatial radius over which to aggregate a match cost function; that is, it limits the negative effects that can result from including match cost information that is too far from the pixel at the center of the aggregation window. The spectral weight term acts as a rudimentary object detector by assigning a higher weight to those pixels that have the same/similar colour. Our analysis indicates that this weighting term is highly useful in creating a high quality spatial aggregate.

In our second study on the components of global stereopsis algorithms we conduct a large scale comparison of the effects on algorithm accuracy obtained by changing only the colour representation used in the input images to the algorithm. We compare the relative accuracy changes observed by using 30 colour representations of a large data set consisting of synthetic and photographic image pairs for input to 75 global stereopsis algorithms formed by combining five global stereopsis frameworks and 15 match cost functions

In our analysis of the data from this second study we determine that this simple change in input colour space, that most researchers should be able to incorporate into their own implementations in five to ten minutes, can potentially yield some significant improvements in the accuracy of a global stereopsis algorithm. We show that utilizing a perceptually uniform colour space, such as CIELuv or CIELuv, in the input to a global stereopsis algorithm can yield a median improvement over using the RGB space for input of as much as 7.2% for some algorithms. Furthermore, we show that algorithms that incorporate a smoothing term that calculates its value by applying a threshold to the gradient magnitude in the colour between neighbouring pixels, and those that incorporate a match cost function similar in style to the L1 or Euclidean distance metrics stand to gain the most by simply switching the colour space of the input to a perceptually uniform colour space.

Finally, we include a plethora of data from both of our studies in the appendices of this dissertation for an interested reader to peruse at their leisure. This data can be used by global stereopsis algorithm researchers and developers to guide some of the choices in their own algorithm development.

7.1 Future Work

This dissertation represents only the tip of a very large iceberg. We have answered some of the "what" questions regarding global stereopsis algorithm components, but there are still many more components to consider as well as the inevitable questions on why a given component works better than some other component under certain circumstances. In the future, we intend to continue chipping away at exposing more of this proverbial iceburg.

7.1.1 Understanding Global Stereopsis

With regard to investigating additional components of global stereopsis algorithms, there are still many more match cost functions that remain to be investigated. There are also large scale comparisons of optimization methods, smoothing assumptions, and additional terms that might be added to an energy equation for global stereopsis waiting to be designed and carried out. Furthermore, a topic not touched in this dissertation is that of post-processing the output of a stereopsis algorithm. There are many different methods of post-processing a disparity map to identify erroneous disparity assignments and/or to make the map more accurate. However, no studies have been conducted to date on which ones perform better and when.

One item that became abundantly clear while trying to analyze the results of the studies in this dissertation is the need for some set of identifying scene characteristics that can be used to identify under what conditions different stereopsis algorithms, or algorithm components, perform better. In the future, we intend to investigate whether such a set of scene characteristics can be developed and automatically detected in a scene. Such a set of characteristics would not only make it possible to begin investigating why different stereopsis algorithms, and their components, perform better on different scenes, but they could also make it possible to create dynamic stereopsis algorithms that automatically alter their own components to optimize their accuracy for a given input.

7.1.2 Colour Representations for Computer Vision

We also intend to continue investigating the use of colour spaces other than RGB in stereopsis algorithms. One route that we plan to investigate is whether the CIELuv and CIELab colour spaces can be made more perceptually uniform than they currently are via a transformation, and whether such a transform would benefit stereopsis algorithms further. Another route we intend to investigate is whether other spaces that we did not consider in our study on colour spaces can benefit global stereopsis algorithm accuracy. We would also like to further investigate the phenomenon that we observed where the HSV colour space yields an accuracy improvement, over RGB, for match cost functions that use the sum minus max channel aggregate; it is possible that a colour space that represents a colour using only its saturation and brightness may be useful in stereopsis algorithms. Finally, we intend to further investigate the use of the Karhunen-Loève transform in creating transformations of colour spaces with the goal of creating a transformation that results in a colour space better suited to stereopsis algorithms.

7.1.3 User-Parameter Optimization

One of the drawbacks of the objective function that we utilize in our response-surface based user-parameter optimization method is that it requires the ground truth disparity map for the image that we are optimizing parameters for. This drawback is inconsequential in the context of algorithm comparison, but makes the objective function unusable for tuning algorithm parameters outside of the research realm; where ground truth is not going to be available. We would like to investigate whether an objective function can be developed that does not require ground truth disparity maps and whose use still yields high quality parameter settings comparable to those obtained when ground truth is available. If such an objective function could be developed then not only could stereopsis algorithm parameters be automatically tuned "in the field," but it could then be possible to develop real-time stereopsis algorithms that continuously, and automatically, tune their parameters while running.

7.1.4 Developing Stereopsis Algorithms

Of course, all of this work developing an understanding of global stereopsis algorithm components would be for naught if we were not going to apply it to actual algorithm development. In the future, we will be applying the knowledge gained from the studies in this dissertation, and any future studies we perform, to develop highly accurate state of the art algorithms for stereopsis.

Bibliography

[1] Motilal Agrawal and Larry S. Davis. Window-based, discontinuity preserving stereo. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages I–66 – I–73, 2004.

[2] Nello Balossino, Maurizio Lucenteforte, Luca Piovano, and Giuseppe Pettiti. A new stereo algorithm integrating luminance, gradient and segmentation informations in a belief-propagation framework. In *International Conference on Image Analysis and Processing*, pages 757 – 762, 2007.

[3] Adrian Barbu and Song-Chun Zhu. Generalizing Swendsen-Wang to sampling arbitrary posterior probabilities. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8):1239 – 1253, 2005.

[4] Herbert Bay, Vittorio Ferraris, and Luc Van Gool. Wide-baseline stereo matching with line segments. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 329 – 336 vol. 1, 2005.

[5] Dinkar N. Bhat and Shree K. Nayar. Ordinal measures for visual correspondence. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 351 – 357, 1996.

[6] Stan Birchfield and Carlo Tomasi. A pixel dissimilarity measure that is insensitive to image sampling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(4):401–406, April 1998.

[7] Michael J. Black and Anand Rangarajan. On the unification of line processes, outlier rejection, and robust statistics with applications in early vision. *International Journal of Computer Vision*, 19(1):57 – 91, 1996.

[8] Aaron Bobick and Stephen Intille. Large occlusion stereo. *International Journal of Computer Vision*, 33(3):181 – 200, September 1999.

[9] Stephen P. Boyd. *Convex Optimization.* Cambridge University Press, 2006.

[10] Yuri Boykov and Vladimir Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(9):1124–1137, 2004.

[11] Yuri Boykov, Olga Veksler, and Ramin Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(11):1222–1239, 2001.

[12] Derek Bradley, Tiberiu Popa, Alla Sheffer, Wolfgang Heidrich, and Tamy Boubekeur. Markerless garment capture. *ACM Transactions on Graphics*, 27(3), 2008.

[13] Douglas C. Brockelbank and Yee-Hong Yang. An experimental investigation in the use of color in computational stereopsis. *IEEE Transactions on Systems, Man, and Cybernetics*, 19(6):1365 – 1383, 1989.

[14] Alan Brunton, Chang Shu, and Gerhard Roth. Belief propagation on the GPU for stereo vision. In *Canadian Conference on Computer and Robot Vision*, pages 76 – 76, 2006.

[15] Xuanping Cai, Dongxiang Zhou, Ganhua Li, and Zhaowen Zhuang. A stereo matching algorithm based on color segments. In *International Conference on Intelligent Robots and Systems*, pages 3372 – 3377, 2005.

[16] Neill W. Campbell, William P. J. Mackeown, Barry T. Thomas, and Tom Troscianko. Interpreting image databases by region classification. *Pattern Recognition*, 30(4):555–563, 1997.

[17] Dengfeng Chai and Qunsheng Peng. Bilayer stereo matching. In *International Conference on Computer Vision*, 2007.

[18] Huahua Chen. Stereo matching using dynamic programming based on occlusion detection. In *International Conference on Mechatronics and Automation*, pages 2445 – 2449, 2007.

[19] Qian Chen and G'erard Medioni. A volumetric stereo matching method: Application to image-based modeling. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 29–34, 1999.

[20] Li Cheng and Terry Caelli. Bayesian stereo matching. In *IEEE Conference on Computer Vision and Pattern Recognition Workshop*, pages 192 – 192, 2004.

[21] Aristide Chikando and Jason Kinser. Optimizing image segmentation using color model mixtures. In *Applied Imagery and Pattern Recognition Workshop*, 2005.

[22] Robert T. Collins. A space-sweep approach for true multi-image matching. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 358–363, 1996.

[23] Dorin Comaniciu and Peter Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5):603 – 619, 2002.

[24] Giovanni Dainese, Marco Marcon, Augusto Sarti, and Stefano Tubaro. 3D object modeling with a voxelset carving approach. In *International Conference on Image Processing*, volume 1, pages I–1053 – I–1056, 2005.

[25] Juan José de Dios and Narciso Garcia. Face detection based on a new color space YCgCr. In *International Conference on Image Processing*, volume 3, pages 909–12, 2003.

[26] H. E. M. den Ouden, R. van Ee, and E. H. F. de Haan. Colour helps to solve the binocular matching problem. *Journal of Physiology*, 567(2):665–671, 2005.

[27] Yi Deng, Qiong Yang, Xueyin Lin, and Xiaoou Tang. Stereo correspondence with occlusion handling in a symmetric patch-based graph-cuts model. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(6):1068 – 1079, 2007.

[28] Guilherme N. DeSouza and Avinash C. Kak. Vision for mobile robot navigation: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(2):237 – 267, 2002.

[29] Laurence S. Dooley, Gour C. Karmakar, and Manzur Murshed. A fuzzy rule-based colour image segmentation algorithm. In *International Conference on Image Processing*, volume 1, pages 977–80, 2003.

[30] Edwin D. El-Mahassni. New robust matching cost functions for stereo vision. In *Conference of the Australian Pattern Recognition Society on Digital Image Computing Techniques and Applications*, pages 144 – 150, 2007.

[31] Mark D. Fairchild. *Color Appearance Models*. IS&T Series in imaging science and technology. Wiley, 2005.

[32] Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Efficient belief propagation for early vision. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages I–261 – I–268, 2004.

[33] Sven Forstmann, Yutaka Kanou, Jun Ohya, Sven Thuering, and Alfred Schmitt. Real-time stereo by using dynamic programming. In *IEEE Conference on Computer Vision and Pattern Recognition Workshop*, pages 29 – 29, 2004.

[34] Stefan K. Gehrig and Uwe Franke. Improving stereo sub-pixel accuracy for long range stereo. In *International Conference on Computer Vision*, 2007.

[35] Theo Gevers and Harro Stokman. Reflectance based edge classification. In *Vision Interface*, pages 25–32, 1999.

[36] Michael Goesele, Brian Curless, and Steven M. Seitz. Multi–view stereo revisited. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 2402 – 2409, 2006.

[37] Bastian Goldlücke and Marcus A. Magnor. Joint 3D-reconstruction and background separation in multiple views using graph cuts. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages I–683 – I–688, 2003.

[38] Minglun Gong and Yee-Hong Yang. Multi-resolution stereo matching using genetic algorithm. In *IEEE Workshop on Stereo and Multi-Baseline Vision*, pages 21–29, 2001.

[39] Minglun Gong and Yee-Hong Yang. Fast stereo matching using reliability-based dynamic programming and consistency constraints. In *International Conference on Computer Vision*, volume 1, pages 610 – 617, 2003.

[40] Minglun Gong and Yee-Hong Yang. Fast unambiguous stereo matching using reliability-based dynamic programming. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(6):998 – 1003, June 2005.

[41] Minglun Gong and Yee-Hong Yang. Near real-time reliable stereo matching using programmable graphics hardware. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 924 – 931, 2005.

[42] Allan Gottschalk and Gershon Buchsbaum. Information theoretic aspects of color signal processing in the visual system. *IEEE Transactions on Systems, Man, and Cybernetics*, 13(5):864 – 873, 1993.

[43] Eric Hamilton. JPEG file interchange format, Version 1.02, September 1992. http://www.jpeg.org/public/jfif.pdf.

[44] Glenn Healey and Raghava Kondepudy. Radiometric CCD camera calibration and noise estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(3):267 – 276, 1994.

[45] Heiko Hirschmüller. Stereo processing by semiglobal matching and mutual information. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(2):328 – 341, 2008.

[46] Heiko Hirschmüller and Daniel Scharstein. Evaluation of cost functions for stereo matching. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[47] Li Hong and George Chen. Segment-based stereo matching using graph cuts. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages I–74 – I–81, 2004.

[48] Xijun Hua, Masahiro Yokomichi, and Michio Kono. Stereo correspondence using color based on competitive-cooperative neural networks. In *International Conference on Parallel and Distributed Computing, Applications and Technologies*, pages 856 – 860, 2005.

[49] Leo M. Hurvich and Dorothea Jameson. An opponent-process theory of color vision. *Psychological Review*, 64(6, Pt.1):384–404, 1957.

[50] P.-M. Jodoin and M. Mignotte. An energy-based framework using global spatial constraints for the stereo correspondence problem. In *International Conference on Image Processing*, volume 5, pages 3001 – 3004, 2004.

[51] John R. Jordan III and Alan C. Bovik. Computational stereo vision using color. *IEEE Control Systems Magazine*, 8(3):31 – 36, 1988.

[52] John R. Jordan III, Wilson S. Geisler, and Alan C. Bovik. Color as a source of information in the stereo correspondence process. *Vision Research*, 30(12):1955–1970, 1990.

[53] Olivier Juan and Yuri Boykov. Active graph cuts. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 1023 – 1029, 2006.

[54] Bahador Khaleghi, Seyed Mohammad Ali Shahabi, and Ali Bidabadi. Performace evaluation of similarity metrics for stereo corresponce problem. In *Canadian Conference on Electrical and Computer Engineering*, pages 1476 – 1478, 2007.

[55] Andreas Klaus, Mario Sormann, and Konrad Karner. Segment–based stereo matching using belief propagation and a self–adapting dissimilarity measure. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 3, pages 15 – 18, 2006.

[56] Pushmeet Kohli and Philip H.S. Torr. Efficiently solving dynamic markov random fields using graph cuts. In *International Conference on Computer Vision*, volume 2, pages 922 – 929, 2005.

[57] Vladimir Kolmogorov and Ramin Zabih. Computing visual correspondence with occlusions using graph cuts. In *International Conference on Computer Vision*, volume 2, pages 508 – 515, 2001.

[58] Vladimir Kolmogorov and Ramin Zabih. Multi-camera scene reconstruction via graph cuts. In *European Conference on Computer Vision*, pages 82 – 96, 2002.

[59] Vladimir Kolmogorov and Ramin Zabih. What energy functions can be minimized via graph cuts? In *European Conference on Computer Vision*, pages 65–81, 2002.

[60] Vladimir Kolmogorov, Ramin Zabih, and Steven Gortler. Generalized multi-camera scene reconstruction using graph cuts. In *Energy Minimization Methods in Computer Vision and Pattern Recognition*, pages 501–516, 2003.

[61] Andreas Koschan. Dense stereo correspondence using polychromatic block matching. In *International Conference on Computer Analysis of Images and Patterns*, pages 538–542, 1993.

[62] Sang Hwa Lee, Jong Il Park, and Choong Woong Lee. A new stereo matching algorithm based on Bayesian model. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 5, pages 2769 – 2772, 1998.

[63] Cheng Lei, Jason Selzer, and Yee-Hong Yang. Region–tree based stereo using dynamic programming optimization. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 2378 – 2385, 2006.

[64] Gang Li and Steven W. Zucker. Surface geometric constraints for stereo in belief propagation. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 2355 – 2362, 2006.

[65] Daniel Lizotte. *Practical Bayesian Optimization*. PhD thesis, University of Alberta, 2008.

[66] David Martin, Charless Fowlkes, Doron Tal, and Jitendra Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *IEEE International Conference on Computer Vision*, pages 416–423, 2001.

[67] Stefano Mattoccia, Federico Tombari, and Luigi Di Stefano. Stereo vision enabling precise border localization within a scanline optimization framework. In *Asian Conference on Computer Vision*, pages 517–527, 2007.

[68] Payman Moallem and Karim Faez. Fast edge-based stereo matching algorithm based on search space reduction. In *IEEE Workshop on Neural Networks for Signal Processing*, pages 587 – 596, 2002.

[69] Daniel Neilson and Yee-Hong Yang. Evaluation of constructable match cost measures for stereo correspondence using cluster ranking. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[70] Noboru Ohta and Alan R. Robertson. *Colorimetry: Fundamentals and Applications*. IS&T Series in imaging science and technology. Wiley, 2005.

[71] Yu-Ichi Ohta, Takeo Kanade, and Toshiyuki Sakai. Color information for region segmentation. *Computer Graphics and Image Processing*, 13(3):222 – 241, July 1980.

[72] Emanuel Parzen. On estimation of a probability density function and mode. *The Annals of Mathematical Statistics*, 33(3):1065–1076, 1962.

[73] George Paschos. Perceptually uniform color spaces for color texture analysis: An empirical evaluation. *IEEE Transactions on Image Processing*, 10(6):932 – 937, 2001.

[74] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, revised second edition, 1988.

[75] Matt Pharr and Grey Humphreys. *Physically Based Rendering: From Theory To Implementation*. Morgan Kaufmann, 2004.

[76] Eran D. Pinhasov, Nahum Shimkin, and Yehoshua Y. Zeevi. Optimal usage of color for disparity estimation in stereo vision. In *The European Signal Processing Conference*, 2005.

[77] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2006.

[78] Kiyohide Satoh and Yuichi Ohta. Occlusion detectable stereo-systematic comparison of detection algorithms. In *International Conference on Pattern Recognition*, volume 1, pages 280 – 286, 1996.

[79] Daniel Scharstein and Chris Pal. Learning conditional random fields for stereo. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[80] Daniel Scharstein and Richard Szeliski. Middlebury stereo vision research web site. http://vision.middlebury.edu/stereo.

[81] Daniel Scharstein and Richard Szeliski. Stereo matching with non–linear diffusion. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 343–350, 1996.

[82] Daniel Scharstein and Richard Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision*, 47(1–3):7–42, 2002.

[83] Daniel Scharstein and Richard Szeliski. High-accuracy stereo depth maps using structured light. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2003.

[84] David Sheskin. *Handbook of Parametric and Nonparametric Statistical Procedures*. CRC Press, 2004.

[85] Steven K. Shevell. *The Science of Color*. Elsevier, second edition, 2003.

[86] David R. Simmons and Frederick A. A. Kingdom. On the independence of chromatic and achromatic stereopsis mechanisms. *Vision Research*, 37(10):1271–1280, 1997.

[87] David R. Simmons and Frederick A. A. Kingdom. Interactions between chromatic- and luminance-contrast-sensitive stereopsis mechanisms. *Vision Research*, 42(12):1535–1545, 2002.

[88] Michael Stokes and Matthew Anderson. A standard default color space for the internet – sRGB, November 1996. http://www.w3.org/Graphics/Color/sRGB.html.

[89] Jian Sun, Nan-Ning Zheng, and Heung-Yeung Shum. Stereo matching using belief propagation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(7):787 – 800, July 2003.

[90] Marshall F. Tappen and William T. Freeman. Comparison of graph cuts with belief propagation for stereo, using identical MRF parameters. In *International Conference on Computer Vision*, volume 2, pages 900 – 906, 2003.

[91] Federico Tombari, Stefano Mattoccia, Luigi De Stefano, and Elisa Addimanda. Classification and evaluation of cost aggregation methods for stereo correspondence. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[92] Federico Tombari, Stefano Mattoccia, and Luigi Di Stefano. Segmentation-based adaptive support for accurate stereo correspondence. In *Proceedings of the Pacific Rim Symposium on Image and Video Technology*, 2007.

[93] Yanghai Tsin, Sing Bing Kang, and Richard Szeliski. Stereo matching with reflections and translucency. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages I–702 – I–709, 2003.

[94] G. Vogiatzis, P.H.S. Torr, and R. Cipolla. Multi-view stereo via volumetric graph-cuts. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 391– 398, 2005.

[95] Radim Šára. Finding the largest unambiguous component of stereo matching. In *European Conference on Computer Vision*, volume 2, pages 900–914, 2002.

[96] Wei Wei and King Ngi Ngan. Disparity estimation with edge-based matching and interpolation. In *IEEE International Symposium on Intelligent Signal Processing and Communication Systems*, pages 153 – 156, 2005.

[97] Yichen Wei and Long Quan. Asymmetrical occlusion handling using graph cut for multi-view stereo. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 902 – 909, 2005.

[98] Slawo Wesolkowski, M.E. Jernigan, and Robert D. Dony. Comparison of color image edge detectors in multiple color spaces. In *International Conference on Image Processing*, volume 2, pages 796 – 799, 2000.

[99] Western Canada Research Grid. Glacier cluster. http://www.westgrid.ca.

[100] Western Canada Research Grid. Snowpatch cluster. http://www.westgrid.ca.

[101] Western Canada Research Grid. Terminus cluster. http://www.westgrid.ca.

[102] Lance Williams. Pyramidal parametrics. In *ACM SIGGRAPH*, pages 1 – 11, 1983.

[103] Günther Wyszecki and W.S. Stiles. *Color Science: Concepts and methods, quantitative data and formulae.* Wiley-Interscience, second edition, 2000.

[104] Qìngxióng Yáng, Liang Wang, Ruigang Yang, Henrik Stewénius, and David Nistér. Stereo matching with color-weighted correlation, hierachical belief propagation and occlusion handling. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 2347 – 2354, 2006.

[105] Kuk-Jin Yoon and In So Kweon. Adaptive support-weight approach for correspondence search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(4):650 – 656, 2006.

[106] Kuk-Jin Yoon and In So Kweon. Stereo matching with symmetric cost functions. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 2371 – 2377, 2006.

[107] Hongsheng Zhang and Shahriar Negahdaripour. BC & GC-based dense stereo by belief propagation. In *IEEE International Conference on Computer Vision Systems*, 2006.

[108] Li Zhang and Steven M. Seitz. Estimating optimal parameters for MRF stereo from a single image pair. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(2):331 – 342, 2007.

[109] Wei Zhang, Quanbing Zhang, Lei Qu, and Sui Wei. A stereo matching algorithm based on multiresolution and epipolar constraint. In *International Conference on Image and Graphics*, pages 180 – 183, 2004.

[110] Dongxiang Zhou, Qiongyu Wu, Yunhui Liu, and Xuanping Cai. A stereo matching algorithm based on fuzzy identification. In *International Conference on Robotics, Intelligent Systems and Signal Processing*, volume 2, pages 1108 – 1113, 2003.

[111] Todd Zickler, Satya P. Mallick, David J. Kriegman, and Peter N. Belhumeur. Color subspaces as photometric invariants. *International Journal of Computer Vision*, 79(1):13–30, 2008.

Appendix A

Data Set

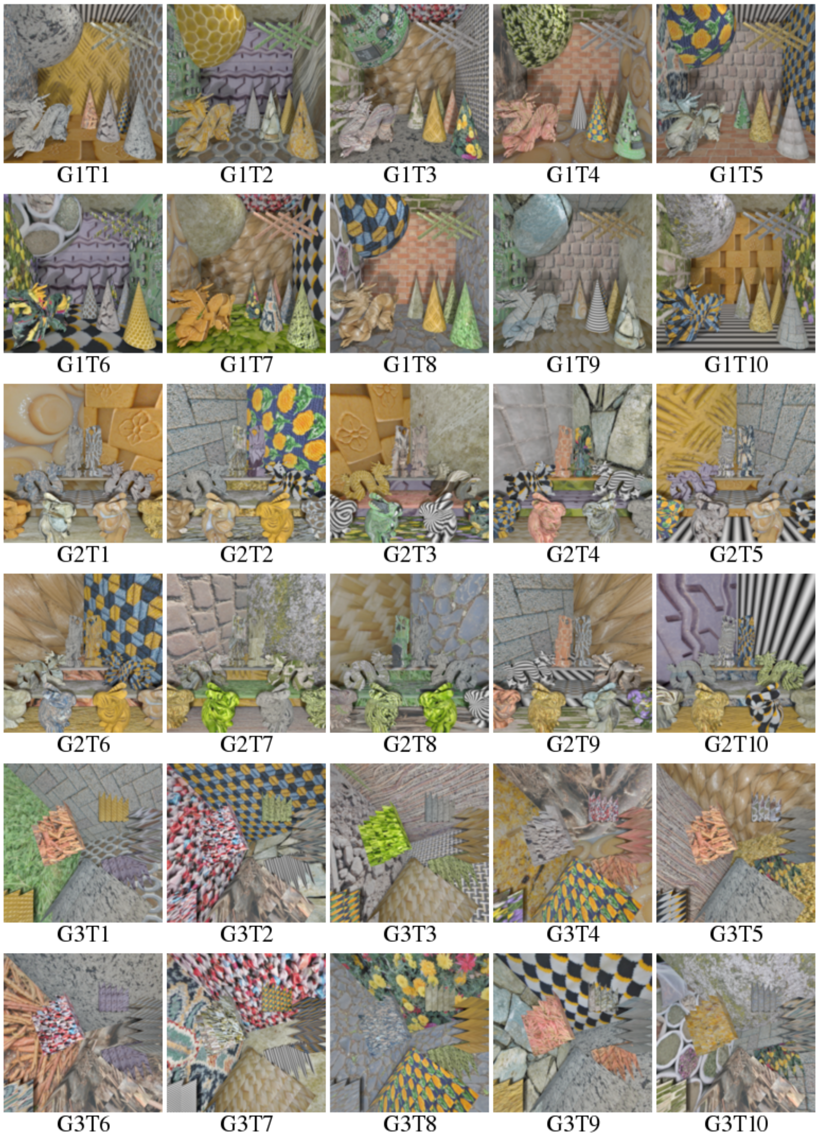
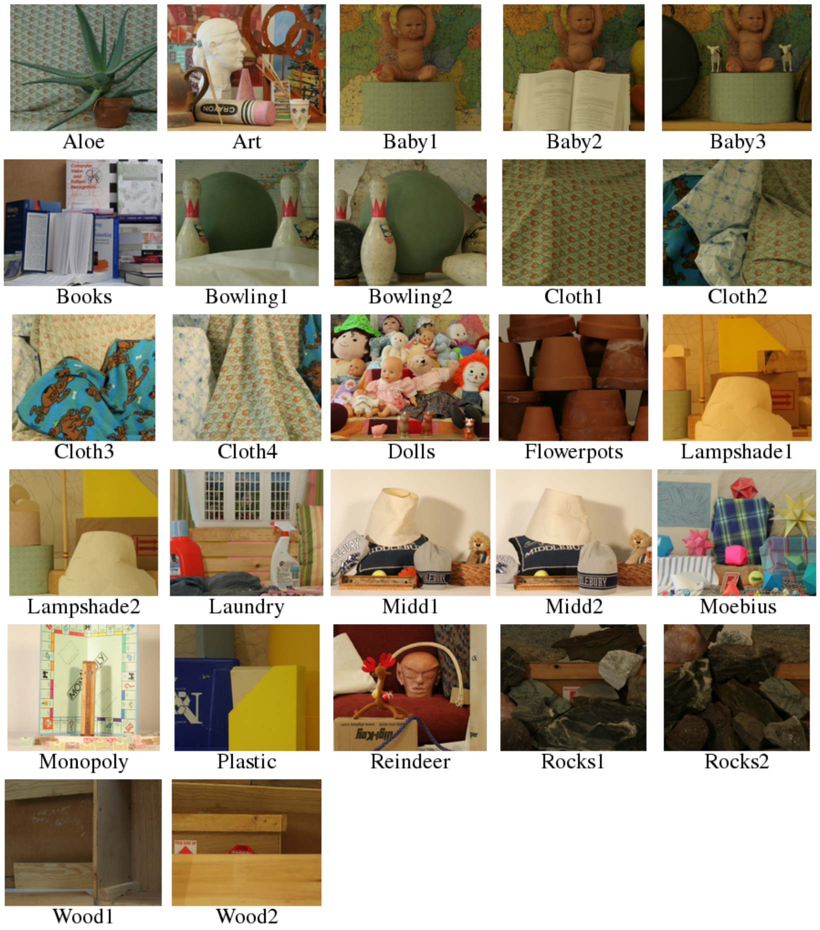


Figure A.1: Synthetic images. All images shown are the left image from the stereo pair.



Aloe Art Baby1 Baby2 Baby3

Books Bowling1 Bowling2 Cloth1 Cloth2

Cloth3 Cloth4 Dolls Flowerpots Lampshade1

Lampshade2 Laundry Midd1 Midd2 Moebius

Monopoly Plastic Reindeer Rocks1 Rocks2

Wood1 Wood2

Figure A.2: Photographic images. All images shown are the left image from the stereo pair.

Appendix B

Truncation Form Ranking Tables

Ranking	**MxAB**	**MxA[t]B**
Over all	2 / 2 / 2	1 / 1 / 1
SO1	2 / 1 / 1	1 / 1 / 1
SO2	1 / 1 / 1	1 / 1 / 1
RTree	2 / 1 / 1	1 / 1 / 1
HBPTL	2 / 2 / 2	1 / 1 / 1
HBPG	2 / 1 / 2	1 / 1 / 1

Table B.1: Rankings of truncation forms for match cost function MxAB at 95% confidence. Ranks listed are for each data set, given in order of increasing noise magnitude.

Ranking	**MxAD**	**MxA[t]D**
Over all	2 / 1 / 1	1 / 1 / 2
SO1	1 / 1 / 1	1 / 1 / 2
SO2	2 / 1 / 1	1 / 1 / 2
RTree	1 / 1 / 1	1 / 1 / 1
HBPTL	2 / 2 / 1	1 / 1 / 1
HBPG	2 / 2 / 2	1 / 1 / 1

Table B.2: Rankings of truncation forms for match cost function MxAD at 95% confidence. Ranks listed are for each data set, given in order of increasing noise magnitude.

Ranking	**MxGB**	**MxG[t]B**
Over all	2 / 1 / 1	1 / 1 / 2
SO1	2 / 1 / 1	1 / 1 / 2
SO2	2 / 1 / 1	1 / 1 / 1
RTree	1 / 1 / 1	1 / 1 / 1
HBPTL	2 / 1 / 1	1 / 1 / 2
HBPG	2 / 1 / 1	1 / 1 / 1

Table B.3: Rankings of truncation forms for match cost function MxGB at 95% confidence. Ranks listed are for each data set, given in order of increasing noise magnitude.

Ranking	**MxGD**	**MxG[t]D**
Over all	2 / 1 / 1	1 / 1 / 2
SO1	2 / 1 / 1	1 / 1 / 2
SO2	1 / 1 / 1	1 / 1 / 2
RTree	1 / 1 / 1	1 / 1 / 1
HBPTL	1 / 1 / 1	1 / 1 / 1
HBPG	2 / 1 / 1	1 / 1 / 1

Table B.4: Rankings of truncation forms for match cost function MxGD at 95% confidence. Ranks listed are for each data set, given in order of increasing noise magnitude.

Ranking	**MxLB**	**MxL[t]B**
Over all	2 / 2 / 2	1 / 1 / 1
SO1	2 / 1 / 1	1 / 1 / 1
SO2	2 / 2 / 1	1 / 1 / 1
RTree	1 / 1 / 1	1 / 1 / 1
HBPTL	2 / 2 / 2	1 / 1 / 1
HBPG	2 / 2 / 2	1 / 1 / 1

Table B.5: Rankings of truncation forms for match cost function MxLB at 95% confidence. Ranks listed are for each data set, given in order of increasing noise magnitude.

Ranking	**MxLD**	**MxL[t]D**
Over all	2 / 1 / 1	1 / 1 / 1
SO1	1 / 1 / 1	1 / 1 / 1
SO2	1 / 1 / 1	2 / 1 / 1
RTree	2 / 1 / 1	1 / 1 / 1
HBPTL	2 / 1 / 1	1 / 1 / 1
HBPG	2 / 1 / 1	1 / 1 / 1

Table B.6: Rankings of truncation forms for match cost function MxLD at 95% confidence. Ranks listed are for each data set, given in order of increasing noise magnitude.

Ranking	**MxSB**	**MxS[t]B**
Over all	2 / 2 / 2	1 / 1 / 1
SO1	2 / 1 / 2	1 / 1 / 1
SO2	2 / 2 / 1	1 / 1 / 1
RTree	2 / 2 / 1	1 / 1 / 1
HBPTL	2 / 2 / 2	1 / 1 / 1
HBPG	2 / 1 / 1	1 / 1 / 1

Table B.7: Rankings of truncation forms for match cost function MxSB at 95% confidence. Ranks listed are for each data set, given in order of increasing noise magnitude.

Ranking	**MxSD**	**MxS[t]D**
Over all	2 / 2 / 2	1 / 1 / 1
SO1	2 / 1 / 1	1 / 1 / 1
SO2	2 / 1 / 2	1 / 1 / 1
RTree	1 / 1 / 1	1 / 1 / 1
HBPTL	2 / 2 / 2	1 / 1 / 1
HBPG	2 / 2 / 2	1 / 1 / 1

Table B.8: Rankings of truncation forms for match cost function MxSD at 95% confidence. Ranks listed are for each data set, given in order of increasing noise magnitude.

Ranking	**SAB**	**SA[t]B**	**S[t]AB**	**S[t]A[t2]B**
Over all	2 / 2 / 3	1 / 1 / 1	1 / 1 / 1	1 / 1 / 2
SO1	2 / 1 / 1	1 / 1 / 1	1 / 2 / 1	1 / 2 / 2
SO2	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 2
RTree	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
HBPTL	2 / 3 / 2	1 / 2 / 1	1 / 1 / 1	1 / 1 / 1
HBPG	2 / 2 / 2	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1

Table B.9: Rankings of truncation forms for match cost function SAB at 95% confidence. Ranks listed are for each data set, given in order of increasing noise magnitude.

Ranking	**SAD**	**SA[t]D**	**S[t]AD**	**S[t]A[t2]D**
Over all	2 / 1 / 1	1 / 1 / 2	1 / 2 / 3	1 / 2 / 4
SO1	1 / 1 / 1	1 / 1 / 2	1 / 2 / 3	1 / 2 / 3
SO2	2 / 1 / 1	1 / 1 / 2	1 / 2 / 3	1 / 2 / 4
RTree	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
HBPTL	2 / 1 / 1	1 / 1 / 1	1 / 1 / 2	1 / 1 / 2
HBPG	2 / 2 / 2	1 / 1 / 1	1 / 1 / 2	1 / 1 / 2

Table B.10: Rankings of truncation forms for match cost function SAD at 95% confidence. Ranks listed are for each data set, given in order of increasing noise magnitude.

Ranking	**SGB**	**SG[t]B**	**S[t]GB**	**S[t]G[t2]B**
Over all	2 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
SO1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 2	1 / 1 / 2
SO2	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
RTree	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
HBPTL	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
HBPG	2 / 2 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1

Table B.11: Rankings of truncation forms for match cost function SGB at 95% confidence. Ranks listed are for each data set, given in order of increasing noise magnitude.

Ranking	**SGD**	**SG[t]D**	**S[t]GD**	**S[t]G[t2]D**
Over all	2 / 2 / 2	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
SO1	2 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
SO2	1 / 2 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
RTree	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
HBPTL	2 / 2 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
HBPG	3 / 2 / 2	2 / 1 / 1	1 / 1 / 1	1 / 1 / 1

Table B.12: Rankings of truncation forms for match cost function SGD at 95% confidence. Ranks listed are for each data set, given in order of increasing noise magnitude.

Ranking	**SLB**	**SL[t]B**	**S[t]LB**	**S[t]L[t2]B**
Over all	2 / 3 / 3	1 / 2 / 2	1 / 1 / 1	1 / 1 / 1
SO1	2 / 2 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
SO2	2 / 2 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
RTree	2 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
HBPTL	2 / 3 / 3	1 / 2 / 2	1 / 1 / 1	1 / 1 / 1
HBPG	2 / 2 / 3	1 / 1 / 2	1 / 1 / 1	1 / 1 / 1

Table B.13: Rankings of truncation forms for match cost function SLB at 95% confidence. Ranks listed are for each data set, given in order of increasing noise magnitude.

Ranking	**SLD**	**SL[t]D**	**S[t]LD**	**S[t]L[t2]D**
Over all	2 / 2 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
SO1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
SO2	1 / 1 / 1	2 / 1 / 1	3 / 1 / 1	3 / 1 / 1
RTree	2 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
HBPTL	2 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
HBPG	2 / 2 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1

Table B.14: Rankings of truncation forms for match cost function SLD at 95% confidence. Ranks listed are for each data set, given in order of increasing noise magnitude.

Ranking	**SSB**	**SS[t]B**	**S[t]SB**	**S[t]S[t2]B**
Over all	3 / 3 / 3	2 / 2 / 2	1 / 1 / 1	1 / 1 / 1
SO1	2 / 2 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
SO2	3 / 2 / 1	2 / 1 / 1	1 / 1 / 1	1 / 1 / 1
RTree	2 / 2 / 2	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
HBPTL	2 / 3 / 2	1 / 2 / 1	1 / 1 / 1	1 / 1 / 1
HBPG	2 / 2 / 2	1 / 2 / 1	1 / 1 / 1	1 / 1 / 1

Table B.15: Rankings of truncation forms for match cost function SSB at 95% confidence. Ranks listed are for each data set, given in order of increasing noise magnitude.

Ranking	**SSD**	**SS[t]D**	**S[t]SD**	**S[t]S[t2]D**
Over all	3 / 2 / 1	2 / 1 / 1	1 / 1 / 1	1 / 1 / 1
SO1	3 / 1 / 1	2 / 1 / 1	1 / 1 / 1	1 / 1 / 1
SO2	3 / 1 / 1	2 / 1 / 1	1 / 1 / 1	1 / 1 / 1
RTree	2 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
HBPTL	3 / 2 / 2	2 / 1 / 1	1 / 1 / 1	1 / 1 / 1
HBPG	3 / 2 / 2	2 / 2 / 1	1 / 1 / 1	1 / 1 / 1

Table B.16: Rankings of truncation forms for match cost function SSD at 95% confidence. Ranks listed are for each data set, given in order of increasing noise magnitude.

Ranking	**SqAB**	**SqA[t]B**	**Sq[t]AB**	**Sq[t]A[t2]B**
Over all	1 / 1 / 1	1 / 1 / 1	2 / 2 / 2	2 / 2 / 2
SO1	1 / 1 / 1	1 / 2 / 1	2 / 3 / 2	2 / 3 / 2
SO2	1 / 1 / 1	1 / 2 / 2	2 / 3 / 3	2 / 3 / 3
RTree	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1	2 / 1 / 1
HBPTL	2 / 1 / 1	1 / 1 / 1	2 / 1 / 1	2 / 1 / 1
HBPG	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1

Table B.17: Rankings of truncation forms for match cost function SqAB at 95% confidence. Ranks listed are for each data set, given in order of increasing noise magnitude.

Ranking	**SqAD**	**SqA[t]D**	**Sq[t]AD**	**Sq[t]A[t2]D**
Over all	2 / 1 / 1	1 / 1 / 2	3 / 2 / 3	3 / 2 / 4
SO1	1 / 1 / 1	1 / 1 / 2	2 / 2 / 3	2 / 2 / 3
SO2	1 / 1 / 1	1 / 2 / 2	2 / 3 / 3	2 / 3 / 4
RTree	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1	2 / 1 / 1
HBPTL	2 / 1 / 1	1 / 1 / 1	3 / 2 / 2	3 / 2 / 2
HBPG	2 / 2 / 1	1 / 1 / 1	2 / 3 / 2	3 / 3 / 2

Table B.18: Rankings of truncation forms for match cost function SqAD at 95% confidence. Ranks listed are for each data set, given in order of increasing noise magnitude.

Ranking	**SqGB**	**SqG[t]B**	**Sq[t]GB**	**Sq[t]G[t2]B**
Over all	2 / 1 / 1	1 / 1 / 1	1 / 1 / 2	1 / 1 / 3
SO1	2 / 1 / 1	1 / 1 / 1	1 / 1 / 2	1 / 1 / 2
SO2	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
RTree	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
HBPTL	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
HBPG	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1

Table B.19: Rankings of truncation forms for match cost function SqGB at 95% confidence. Ranks listed are for each data set, given in order of increasing noise magnitude.

Ranking	**SqGD**	**SqG[t]D**	**Sq[t]GD**	**Sq[t]G[t2]D**
Over all	2 / 2 / 2	1 / 1 / 1	1 / 1 / 2	1 / 1 / 2
SO1	2 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
SO2	1 / 1 / 1	1 / 1 / 1	1 / 1 / 2	1 / 1 / 2
RTree	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
HBPTL	2 / 2 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
HBPG	3 / 2 / 1	2 / 2 / 1	1 / 1 / 1	1 / 1 / 1

Table B.20: Rankings of truncation forms for match cost function SqGD at 95% confidence. Ranks listed are for each data set, given in order of increasing noise magnitude.

Ranking	**SqLB**	**SqL[t]B**	**Sq[t]LB**	**Sq[t]L[t2]B**
Over all	3 / 3 / 2	2 / 2 / 1	1 / 1 / 1	1 / 2 / 1
SO1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
SO2	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
RTree	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
HBPTL	2 / 2 / 3	1 / 1 / 2	1 / 1 / 1	1 / 1 / 2
HBPG	2 / 1 / 2	2 / 1 / 1	1 / 1 / 1	1 / 1 / 1

Table B.21: Rankings of truncation forms for match cost function SqLB at 95% confidence. Ranks listed are for each data set, given in order of increasing noise magnitude.

Ranking	**SqLD**	**SqL[t]D**	**Sq[t]LD**	**Sq[t]L[t2]D**
Over all	3 / 1 / 1	2 / 1 / 1	1 / 1 / 1	1 / 1 / 1
SO1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 2 / 1
SO2	2 / 1 / 1	1 / 1 / 1	1 / 1 / 2	1 / 1 / 2
RTree	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
HBPTL	2 / 2 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
HBPG	3 / 2 / 1	2 / 1 / 1	1 / 1 / 1	1 / 1 / 1

Table B.22: Rankings of truncation forms for match cost function SqLD at 95% confidence. Ranks listed are for each data set, given in order of increasing noise magnitude.

Ranking	**SqSB**	**SqS[t]B**	**Sq[t]SB**	**Sq[t]S[t2]B**
Over all	2 / 2 / 2	2 / 2 / 2	1 / 1 / 1	1 / 1 / 1
SO1	1 / 1 / 1	2 / 1 / 1	1 / 1 / 1	1 / 1 / 1
SO2	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
RTree	2 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
HBPTL	2 / 2 / 2	2 / 2 / 2	1 / 1 / 1	1 / 1 / 1
HBPG	2 / 2 / 2	2 / 2 / 2	1 / 1 / 1	1 / 1 / 1

Table B.23: Rankings of truncation forms for match cost function SqSB at 95% confidence. Ranks listed are for each data set, given in order of increasing noise magnitude.

Ranking	**SqSD**	**SqS[t]D**	**Sq[t]SD**	**Sq[t]S[t2]D**
Over all	3 / 2 / 1	2 / 1 / 1	1 / 2 / 2	1 / 2 / 2
SO1	2 / 1 / 1	1 / 1 / 1	1 / 1 / 2	1 / 2 / 2
SO2	2 / 1 / 1	1 / 1 / 1	1 / 2 / 2	1 / 2 / 2
RTree	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
HBPTL	3 / 2 / 1	2 / 1 / 1	1 / 1 / 1	1 / 1 / 1
HBPG	3 / 3 / 2	2 / 2 / 1	1 / 1 / 1	1 / 1 / 1

Table B.24: Rankings of truncation forms for match cost function SqSD at 95% confidence. Ranks listed are for each data set, given in order of increasing noise magnitude.

Ranking	**SwAB**	**SwA[t]B**	**Sw[t]AB**	**Sw[t]A[t2]B**
Over all	2 / 3 / 2	1 / 1 / 1	1 / 1 / 1	1 / 2 / 1
SO1	2 / 2 / 1	1 / 1 / 1	1 / 1 / 1	1 / 2 / 1
SO2	2 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
RTree	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
HBPTL	2 / 2 / 2	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
HBPG	1 / 2 / 2	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1

Table B.25: Rankings of truncation forms for match cost function SwAB at 95% confidence. Ranks listed are for each data set, given in order of increasing noise magnitude.

Ranking	**SwAD**	**SwA[t]D**	**Sw[t]AD**	**Sw[t]A[t2]D**
Over all	2 / 2 / 1	1 / 1 / 1	1 / 1 / 2	1 / 1 / 3
SO1	2 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
SO2	2 / 1 / 1	1 / 1 / 2	1 / 1 / 2	1 / 1 / 3
RTree	1 / 1 / 1	1 / 1 / 1	2 / 1 / 1	1 / 1 / 1
HBPTL	3 / 2 / 1	2 / 1 / 1	1 / 1 / 1	1 / 1 / 1
HBPG	2 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1

Table B.26: Rankings of truncation forms for match cost function SwAD at 95% confidence. Ranks listed are for each data set, given in order of increasing noise magnitude.

Ranking	**SwGB**	**SwG[t]B**	**Sw[t]GB**	**Sw[t]G[t2]B**
Over all	2 / 1 / 2	1 / 1 / 2	1 / 1 / 1	1 / 1 / 2
SO1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 2
SO2	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
RTree	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
HBPTL	3 / 2 / 1	1 / 2 / 1	2 / 1 / 1	2 / 2 / 1
HBPG	2 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1

Table B.27: Rankings of truncation forms for match cost function SwGB at 95% confidence. Ranks listed are for each data set, given in order of increasing noise magnitude.

Ranking	**SwGD**	**SwG[t]D**	**Sw[t]GD**	**Sw[t]G[t2]D**
Over all	2 / 2 / 3	1 / 1 / 1	1 / 1 / 1	1 / 1 / 2
SO1	1 / 1 / 2	1 / 1 / 1	1 / 1 / 1	1 / 1 / 2
SO2	1 / 2 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
RTree	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
HBPTL	2 / 2 / 2	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
HBPG	2 / 2 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1

Table B.28: Rankings of truncation forms for match cost function SwGD at 95% confidence. Ranks listed are for each data set, given in order of increasing noise magnitude.

Ranking	**SwLB**	**SwL[t]B**	**Sw[t]LB**	**Sw[t]L[t2]B**
Over all	2 / 2 / 2	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
SO1	2 / 2 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
SO2	2 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
RTree	2 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
HBPTL	3 / 2 / 2	1 / 1 / 1	1 / 1 / 1	2 / 1 / 1
HBPG	2 / 2 / 2	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1

Table B.29: Rankings of truncation forms for match cost function SwLB at 95% confidence. Ranks listed are for each data set, given in order of increasing noise magnitude.

Ranking	**SwLD**	**SwL[t]D**	**Sw[t]LD**	**Sw[t]L[t2]D**
Over all	2 / 2 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
SO1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
SO2	1 / 1 / 1	2 / 1 / 1	2 / 1 / 1	2 / 1 / 1
RTree	2 / 2 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
HBPTL	2 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
HBPG	3 / 1 / 1	2 / 1 / 1	1 / 1 / 1	1 / 1 / 1

Table B.30: Rankings of truncation forms for match cost function SwLD at 95% confidence. Ranks listed are for each data set, given in order of increasing noise magnitude.

Ranking	**SwSB**	**SwS[t]B**	**Sw[t]SB**	**Sw[t]S[t2]B**
Over all	2 / 2 / 2	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
SO1	2 / 2 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
SO2	2 / 2 / 2	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
RTree	2 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
HBPTL	2 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
HBPG	2 / 2 / 2	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1

Table B.31: Rankings of truncation forms for match cost function SwSB at 95% confidence. Ranks listed are for each data set, given in order of increasing noise magnitude.

Ranking	**SwSD**	**SwS[t]D**	**Sw[t]SD**	**Sw[t]S[t2]D**
Over all	2 / 2 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
SO1	2 / 2 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
SO2	2 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 2
RTree	2 / 2 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
HBPTL	2 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
HBPG	2 / 2 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1

Table B.32: Rankings of truncation forms for match cost function SwSD at 95% confidence. Ranks listed are for each data set, given in order of increasing noise magnitude.

Ranking	**SxAB**	**SxA[t]B**	**Sx[t]AB**	**Sx[t]A[t2]B**
Over all	2 / 2 / 2	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
SO1	2 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
SO2	2 / 1 / 1	1 / 1 / 1	1 / 1 / 1	2 / 1 / 1
RTree	1 / 1 / 2	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
HBPTL	2 / 3 / 2	1 / 2 / 1	1 / 1 / 1	1 / 2 / 1
HBPG	1 / 2 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1

Table B.33: Rankings of truncation forms for match cost function SxAB at 95% confidence. Ranks listed are for each data set, given in order of increasing noise magnitude.

Ranking	**SxAD**	**SxA[t]D**	**Sx[t]AD**	**Sx[t]A[t2]D**
Over all	2 / 2 / 1	1 / 1 / 2	1 / 2 / 3	1 / 3 / 4
SO1	1 / 1 / 1	1 / 1 / 2	1 / 1 / 3	1 / 2 / 3
SO2	2 / 1 / 1	1 / 1 / 2	1 / 2 / 3	1 / 2 / 3
RTree	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
HBPTL	2 / 2 / 1	1 / 1 / 1	1 / 1 / 2	1 / 1 / 2
HBPG	2 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 2

Table B.34: Rankings of truncation forms for match cost function SxAD at 95% confidence. Ranks listed are for each data set, given in order of increasing noise magnitude.

Ranking	**SxGB**	**SxG[t]B**	**Sx[t]GB**	**Sx[t]G[t2]B**
Over all	2 / 2 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
SO1	2 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
SO2	1 / 1 / 2	1 / 1 / 2	1 / 1 / 1	1 / 1 / 2
RTree	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
HBPTL	2 / 1 / 1	1 / 1 / 1	1 / 1 / 2	1 / 1 / 2
HBPG	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1

Table B.35: Rankings of truncation forms for match cost function SxGB at 95% confidence. Ranks listed are for each data set, given in order of increasing noise magnitude.

Ranking	**SxGD**	**SxG[t]D**	**Sx[t]GD**	**Sx[t]G[t2]D**
Over all	2 / 3 / 2	1 / 1 / 1	1 / 1 / 1	1 / 2 / 1
SO1	1 / 2 / 2	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
SO2	1 / 2 / 2	1 / 1 / 2	1 / 1 / 1	1 / 1 / 2
RTree	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
HBPTL	2 / 2 / 1	1 / 1 / 1	1 / 1 / 1	1 / 2 / 1
HBPG	2 / 2 / 2	2 / 1 / 1	1 / 1 / 2	1 / 2 / 2

Table B.36: Rankings of truncation forms for match cost function SxGD at 95% confidence. Ranks listed are for each data set, given in order of increasing noise magnitude.

Ranking	**SxLB**	**SxL[t]B**	**Sx[t]LB**	**Sx[t]L[t2]B**
Over all	2 / 2 / 2	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
SO1	3 / 2 / 1	1 / 1 / 1	1 / 1 / 1	2 / 1 / 1
SO2	2 / 2 / 2	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
RTree	3 / 2 / 1	1 / 1 / 1	2 / 1 / 1	1 / 1 / 1
HBPTL	2 / 2 / 2	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
HBPG	2 / 2 / 2	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1

Table B.37: Rankings of truncation forms for match cost function SxLB at 95% confidence. Ranks listed are for each data set, given in order of increasing noise magnitude.

Ranking	**SxLD**	**SxL[t]D**	**Sx[t]LD**	**Sx[t]L[t2]D**
Over all	1 / 1 / 1	2 / 1 / 1	1 / 1 / 1	1 / 1 / 1
SO1	1 / 1 / 1	2 / 1 / 1	1 / 1 / 1	2 / 1 / 1
SO2	1 / 2 / 1	2 / 1 / 1	2 / 1 / 1	2 / 1 / 1
RTree	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
HBPTL	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
HBPG	2 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1

Table B.38: Rankings of truncation forms for match cost function SxLD at 95% confidence. Ranks listed are for each data set, given in order of increasing noise magnitude.

Ranking	**SxSB**	**SxS[t]B**	**Sx[t]SB**	**Sx[t]S[t2]B**
Over all	3 / 2 / 2	2 / 1 / 1	1 / 1 / 1	2 / 1 / 1
SO1	2 / 2 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
SO2	2 / 2 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
RTree	3 / 2 / 2	2 / 1 / 1	1 / 1 / 1	2 / 1 / 1
HBPTL	2 / 2 / 2	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
HBPG	2 / 3 / 2	1 / 1 / 1	1 / 2 / 1	1 / 1 / 1

Table B.39: Rankings of truncation forms for match cost function SxSB at 95% confidence. Ranks listed are for each data set, given in order of increasing noise magnitude.

Ranking	**SxSD**	**SxS[t]D**	**Sx[t]SD**	**Sx[t]S[t2]D**
Over all	3 / 2 / 1	2 / 1 / 1	1 / 1 / 1	1 / 1 / 1
SO1	3 / 1 / 1	2 / 1 / 1	1 / 1 / 1	1 / 1 / 1
SO2	3 / 1 / 1	2 / 1 / 1	1 / 1 / 1	1 / 1 / 1
RTree	2 / 2 / 2	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
HBPTL	3 / 2 / 1	2 / 1 / 1	1 / 1 / 1	1 / 1 / 1
HBPG	2 / 1 / 1	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1

Table B.40: Rankings of truncation forms for match cost function SxSD at 95% confidence. Ranks listed are for each data set, given in order of increasing noise magnitude.

Appendix C

Channel Function Rankings For Each Global Stereopsis Framework

Noise→	**None**		**Low**		**High**		**Noise→**	**None**		**Low**		**High**		**Noise→**	**None**		**Low**		**High**	
Measure	**B**	**D**	**B**	**D**	**B**	**D**	**Measure**	**B**	**D**	**B**	**D**	**B**	**D**	**Measure**	**B**	**D**	**B**	**D**	**B**	**D**
MxA*	2	1	2	1	1	1	MxA[t]*	2	1	2	1	1	2	MxG*	2	1	2	1	1	1
MxG[t]*	2	1	2	1	1	1	MxL*	2	1	2	1	2	1	MxL[t]*	2	1	2	1	2	1
MxS*	2	1	2	1	2	1	MxS[t]*	2	1	2	1	2	1	SA*	2	1	2	1	1	1
SA[t]*	2	1	2	1	1	1	SG*	2	1	2	1	1	1	SG[t]*	2	1	2	1	1	1
SL*	2	1	2	1	2	1	SL[t]*	2	1	2	1	2	1	SS*	2	1	2	1	2	1
SS[t]*	2	1	2	1	2	1	S[t]A*	2	1	1	2	1	2	S[t]A[t2]*	2	1	1	2	1	2
S[t]G*	2	1	2	1	2	1	S[t]G[t2]*	2	1	2	1	2	1	S[t]L*	1	1	2	1	2	1
S[t]L[t2]*	1	1	2	1	2	1	S[t]S*	2	1	2	1	2	1	S[t]S[t2]*	2	1	2	1	2	1
SqA*	2	1	1	1	1	2	SqA[t]*	2	1	1	1	1	2	SqG*	2	1	2	1	1	1
SqG[t]*	2	1	2	1	1	1	SqL*	2	1	2	1	1	1	SqL[t]*	2	1	2	1	1	1
SqS*	2	1	2	1	1	1	SqS[t]*	2	1	2	1	1	1	Sq[t]A*	1	2	1	2	1	2
Sq[t]A[t2]*	1	2	1	2	1	2	Sq[t]G*	2	1	2	1	1	1	Sq[t]G[t2]*	2	1	2	1	1	1
Sq[t]L*	2	1	2	1	1	1	Sq[t]L[t2]*	2	1	2	1	1	1	Sq[t]S*	2	1	2	1	1	2
Sq[t]S[t2]*	2	1	2	1	1	2	SwA*	2	1	2	1	1	1	SwA[t]*	2	1	2	1	1	1
SwG*	2	1	2	1	2	1	SwG[t]*	2	1	2	1	2	1	SwL*	2	1	2	1	2	1
SwL[t]*	2	1	2	1	2	1	SwS*	2	1	2	1	2	1	SwS[t]*	2	1	2	1	2	1
Sw[t]A*	2	1	2	1	1	1	Sw[t]A[t2]*	2	1	2	1	1	1	Sw[t]G*	2	1	2	1	2	1
Sw[t]G[t2]*	2	1	2	1	2	1	Sw[t]L*	1	1	2	1	2	1	Sw[t]L[t2]*	1	1	2	1	2	1
Sw[t]S*	2	1	2	1	2	1	Sw[t]S[t2]*	2	1	2	1	2	1	SxA*	2	1	2	1	2	1
SxA[t]*	2	1	2	1	2	1	SxG*	2	1	2	1	2	1	SxG[t]*	2	1	2	1	2	1
SxL*	2	1	2	1	2	1	SxL[t]*	2	1	2	1	2	1	SxS*	2	1	2	1	2	1
SxS[t]*	2	1	2	1	2	1	Sx[t]A*	2	1	2	1	1	2	Sx[t]A[t2]*	2	1	2	1	1	2
Sx[t]G*	2	1	2	1	2	1	Sx[t]G[t2]*	2	1	2	1	2	1	Sx[t]L*	2	1	2	1	2	1
Sx[t]L[t2]*	2	1	2	1	2	1	Sx[t]S*	2	1	2	1	2	1	Sx[t]S[t2]*	2	1	2	1	2	1

Table C.1: Over all ranking of channel functions for each match cost function at 95% confidence for each data set in the SO1 framework.

Noise→	**None**		**Low**		**High**		**Noise→**	**None**		**Low**		**High**		**Noise→**	**None**		**Low**		**High**	
Measure	**B**	**D**	**B**	**D**	**B**	**D**	**Measure**	**B**	**D**	**B**	**D**	**B**	**D**	**Measure**	**B**	**D**	**B**	**D**	**B**	**D**
MxA*	2	1	1	1	1	2	MxA[t]*	2	1	2	1	1	2	MxG*	2	1	2	1	1	1
MxG[t]*	2	1	2	1	1	2	MxL*	2	1	2	1	1	2	MxL[t]*	2	1	2	1	1	2
MxS*	2	1	2	1	1	1	MxS[t]*	2	1	2	1	1	1	SA*	2	1	1	1	1	2
SA[t]*	2	1	1	1	1	2	SG*	2	1	2	1	1	1	SG[t]*	2	1	2	1	1	1
SL*	2	1	2	1	1	1	SL[t]*	2	1	2	1	1	1	SS*	2	1	2	1	1	1
SS[t]*	2	1	2	1	1	1	S[t]A*	2	1	1	2	1	2	S[t]A[t2]*	2	1	1	2	1	2
S[t]G*	2	1	2	1	1	1	S[t]G[t2]*	2	1	2	1	1	1	S[t]L*	1	1	2	1	1	1
S[t]L[t2]*	1	1	2	1	1	1	S[t]S*	2	1	2	1	1	1	S[t]S[t2]*	2	1	2	1	1	1
SqA*	2	1	1	1	1	2	SqA[t]*	2	1	1	2	1	2	SqG*	2	1	1	1	1	2
SqG[t]*	2	1	2	1	1	1	SqL*	2	1	1	1	1	2	SqL[t]*	2	1	2	1	1	2
SqS*	2	1	1	1	1	2	SqS[t]*	2	1	1	1	1	2	Sq[t]A*	1	2	1	2	1	2
Sq[t]A[t2]*	1	2	1	2	1	2	Sq[t]G*	2	1	2	1	1	2	Sq[t]G[t2]*	2	1	2	1	1	2
Sq[t]L*	2	1	1	1	1	2	Sq[t]L[t2]*	2	1	2	1	1	2	Sq[t]S*	2	1	1	2	1	2
Sq[t]S[t2]*	2	1	1	2	1	2	SwA*	2	1	1	1	1	2	SwA[t]*	2	1	1	1	1	2
SwG*	2	1	2	1	1	1	SwG[t]*	2	1	2	1	1	1	SwL*	2	1	2	1	2	1
SwL[t]*	1	1	2	1	2	1	SwS*	2	1	2	1	2	1	SwS[t]*	2	1	2	1	2	1
Sw[t]A*	2	1	1	1	1	2	Sw[t]A[t2]*	2	1	1	1	1	2	Sw[t]G*	2	1	2	1	2	1
Sw[t]G[t2]*	2	1	2	1	1	1	Sw[t]L*	1	1	2	1	2	1	Sw[t]L[t2]*	1	1	2	1	1	1
Sw[t]S*	2	1	2	1	2	1	Sw[t]S[t2]*	2	1	2	1	1	1	SxA*	2	1	1	1	1	1
SxA[t]*	2	1	1	1	1	2	SxG*	2	1	2	1	2	1	SxG[t]*	2	1	2	1	2	1
SxL*	2	1	2	1	2	1	SxL[t]*	2	1	2	1	2	1	SxS*	2	1	2	1	2	1
SxS[t]*	2	1	2	1	2	1	Sx[t]A*	2	1	1	2	1	2	Sx[t]A[t2]*	2	1	1	2	1	2
Sx[t]G*	2	1	2	1	2	1	Sx[t]G[t2]*	2	1	2	1	2	1	Sx[t]L*	2	1	2	1	2	1
Sx[t]L[t2]*	2	1	2	1	2	1	Sx[t]S*	2	1	2	1	2	1	Sx[t]S[t2]*	2	1	2	1	2	1

Table C.2: Over all ranking of channel functions for each match cost function at 95% confidence for each data set in the SO2 framework.

Noise→	**None**		**Low**		**High**		**Noise→**	**None**		**Low**		**High**		**Noise→**	**None**		**Low**		**High**	
Measure	**B**	**D**	**B**	**D**	**B**	**D**	**Measure**	**B**	**D**	**B**	**D**	**B**	**D**	**Measure**	**B**	**D**	**B**	**D**	**B**	**D**
MxA*	1	1	1	1	1	1	MxA[t]*	1	1	1	1	1	1	MxG*	1	1	1	1	1	1
MxG[t]*	1	1	1	1	1	1	MxL*	1	1	1	1	1	1	MxL[t]*	1	1	1	1	1	1
MxS*	2	1	2	1	1	1	MxS[t]*	2	1	1	1	1	1	SA*	2	1	1	1	1	1
SA[t]*	1	1	1	1	1	1	SG*	1	1	1	1	1	1	SG[t]*	1	1	1	1	1	1
SL*	2	1	1	1	1	1	SL[t]*	1	1	1	1	1	1	SS*	2	1	1	1	2	1
SS[t]*	1	1	1	1	1	1	S[t]A*	1	1	1	1	1	1	S[t]A[t2]*	1	1	1	1	1	1
S[t]G*	1	1	1	1	1	1	S[t]G[t2]*	1	1	1	1	1	1	S[t]L*	1	1	1	1	1	1
S[t]L[t2]*	1	1	1	1	1	1	S[t]S*	1	1	1	1	1	1	S[t]S[t2]*	1	1	1	1	1	1
SqA*	1	1	1	1	1	1	SqA[t]*	1	1	1	1	1	1	SqG*	1	1	1	1	1	1
SqG[t]*	1	1	1	1	1	1	SqL*	1	1	1	1	1	1	SqL[t]*	1	1	1	1	1	1
SqS*	1	1	1	1	1	1	SqS[t]*	1	1	1	1	1	1	Sq[t]A*	1	1	1	1	1	1
Sq[t]A[t2]*	1	1	1	1	1	1	Sq[t]G*	1	1	1	1	1	1	Sq[t]G[t2]*	1	1	1	1	1	1
Sq[t]L*	1	1	1	1	1	1	Sq[t]L[t2]*	1	1	1	1	1	1	Sq[t]S*	1	1	1	1	1	1
Sq[t]S[t2]*	1	1	1	1	1	1	SwA*	1	1	1	1	1	1	SwA[t]*	1	1	1	1	1	1
SwG*	1	1	1	1	1	1	SwG[t]*	1	1	1	1	1	1	SwL*	2	1	1	1	1	1
SwL[t]*	1	1	1	1	1	1	SwS*	2	1	1	1	1	1	SwS[t]*	2	1	1	1	1	1
Sw[t]A*	1	2	1	1	1	1	Sw[t]A[t2]*	1	1	1	1	1	1	Sw[t]G*	1	1	1	1	1	1
Sw[t]G[t2]*	1	1	1	1	1	1	Sw[t]L*	2	1	1	1	1	1	Sw[t]L[t2]*	1	1	1	1	1	1
Sw[t]S*	2	1	1	1	1	1	Sw[t]S[t2]*	2	1	1	1	1	1	SxA*	2	1	1	1	2	1
SxA[t]*	1	1	1	1	1	1	SxG*	1	1	1	1	1	1	SxG[t]*	1	1	1	1	1	1
SxL*	2	1	1	1	1	1	SxL[t]*	1	1	1	1	1	1	SxS*	2	1	2	1	2	1
SxS[t]*	2	1	1	1	2	1	Sx[t]A*	1	1	1	1	1	1	Sx[t]A[t2]*	1	1	1	1	1	1
Sx[t]G*	1	1	1	1	1	1	Sx[t]G[t2]*	1	1	1	1	1	1	Sx[t]L*	2	1	1	1	1	1
Sx[t]L[t2]*	1	1	1	1	1	1	Sx[t]S*	2	1	1	1	1	1	Sx[t]S[t2]*	2	1	1	1	1	1

Table C.3: Over all ranking of channel functions for each match cost function at 95% confidence for each data set in the RTree framework.

Noise→	**None**		**Low**		**High**		**Noise→**	**None**		**Low**		**High**		**Noise→**	**None**		**Low**		**High**	
Measure	**B**	**D**	**B**	**D**	**B**	**D**	**Measure**	**B**	**D**	**B**	**D**	**B**	**D**	**Measure**	**B**	**D**	**B**	**D**	**B**	**D**
MxA*	2	1	2	1	2	1	MxA[t]*	2	1	1	1	1	1	MxG*	2	1	1	1	1	1
MxG[t]*	2	1	1	1	1	1	MxL*	2	1	2	1	2	1	MxL[t]*	2	1	2	1	1	1
MxS*	1	1	2	1	2	1	MxS[t]*	2	1	2	1	2	1	SA*	2	1	2	1	2	1
SA[t]*	2	1	2	1	1	1	SG*	2	1	1	1	1	1	SG[t]*	2	1	1	1	1	1
SL*	1	1	2	1	2	1	SL[t]*	2	1	2	1	1	1	SS*	1	1	2	1	2	1
SS[t]*	1	1	2	1	2	1	S[t]A*	2	1	1	1	1	2	S[t]A[t2]*	2	1	1	1	1	2
S[t]G*	2	1	1	1	1	2	S[t]G[t2]*	2	1	1	1	1	1	S[t]L*	2	1	1	1	1	1
S[t]L[t2]*	2	1	2	1	1	1	S[t]S*	2	1	2	1	2	1	S[t]S[t2]*	2	1	2	1	2	1
SqA*	2	1	1	1	1	1	SqA[t]*	2	1	1	1	1	1	SqG*	2	1	1	1	1	1
SqG[t]*	2	1	2	1	1	1	SqL*	2	1	2	1	2	1	SqL[t]*	2	1	2	1	1	1
SqS*	2	1	2	1	2	1	SqS[t]*	2	1	2	1	2	1	Sq[t]A*	1	2	1	2	1	2
Sq[t]A[t2]*	1	2	1	2	1	2	Sq[t]G*	2	1	1	1	1	1	Sq[t]G[t2]*	2	1	2	1	1	1
Sq[t]L*	2	1	2	1	1	1	Sq[t]L[t2]*	2	1	2	1	1	1	Sq[t]S*	2	1	2	1	1	1
Sq[t]S[t2]*	2	1	2	1	1	2	SwA*	2	1	2	1	1	1	SwA[t]*	2	1	1	1	1	1
SwG*	2	1	1	1	1	1	SwG[t]*	2	1	1	1	1	1	SwL*	2	1	2	1	2	1
SwL[t]*	1	1	2	1	1	1	SwS*	1	1	2	1	2	1	SwS[t]*	1	1	2	1	2	1
Sw[t]A*	2	1	2	1	1	1	Sw[t]A[t2]*	2	1	2	1	1	1	Sw[t]G*	2	1	1	1	1	2
Sw[t]G[t2]*	2	1	2	1	1	1	Sw[t]L*	1	1	2	1	1	1	Sw[t]L[t2]*	2	1	1	1	1	1
Sw[t]S*	1	1	2	1	2	1	Sw[t]S[t2]*	1	1	2	1	2	1	SxA*	2	1	2	1	2	1
SxA[t]*	2	1	2	1	1	1	SxG*	2	1	1	1	1	1	SxG[t]*	2	1	1	1	1	1
SxL*	2	1	2	1	2	1	SxL[t]*	2	1	2	1	2	1	SxS*	1	1	2	1	2	1
SxS[t]*	1	1	2	1	2	1	Sx[t]A*	2	1	1	1	1	1	Sx[t]A[t2]*	2	1	2	1	1	2
Sx[t]G*	2	1	1	1	2	1	Sx[t]G[t2]*	2	1	1	1	2	1	Sx[t]L*	2	1	2	1	2	1
Sx[t]L[t2]*	2	1	2	1	2	1	Sx[t]S*	1	1	2	1	2	1	Sx[t]S[t2]*	1	1	2	1	2	1

Table C.4: Over all ranking of channel functions for each match cost function at 95% confidence for each data set in the HBPTL framework.

Noise→	None		Low		High		Noise→	None		Low		High		Noise→	None		Low		High	
Measure	**B**	**D**	**B**	**D**	**B**	**D**	**Measure**	**B**	**D**	**B**	**D**	**B**	**D**	**Measure**	**B**	**D**	**B**	**D**	**B**	**D**
MxA*	1	1	1	1	1	1	MxA[t]*	2	1	1	1	1	1	MxG*	1	1	1	1	1	1
MxG[t]*	2	1	1	1	1	1	MxL*	1	1	1	1	2	1	MxL[t]*	2	1	1	1	1	1
MxS*	1	1	1	1	1	1	MxS[t]*	1	1	2	1	2	1	SA*	1	1	1	1	1	1
SA[t]*	1	1	1	1	1	1	SG*	1	1	1	1	1	1	SG[t]*	1	1	1	1	1	1
SL*	1	1	1	1	2	1	SL[t]*	2	1	1	1	1	1	SS*	1	1	2	1	2	1
SS[t]*	1	1	2	1	2	1	S[t]A*	1	1	1	1	1	2	S[t]A[t2]*	2	1	1	1	1	2
S[t]G*	1	1	1	1	1	2	S[t]G[t2]*	1	1	1	1	1	1	S[t]L*	1	1	1	1	1	1
S[t]L[t2]*	1	1	1	1	1	1	S[t]S*	1	1	2	1	2	1	S[t]S[t2]*	1	1	2	1	2	1
SqA*	1	1	1	1	1	1	SqA[t]*	2	1	1	1	1	1	SqG*	1	1	1	2	1	1
SqG[t]*	1	1	1	1	1	1	SqL*	1	1	1	1	1	1	SqL[t]*	2	1	1	1	1	1
SqS*	1	1	1	1	1	1	SqS[t]*	1	1	1	1	1	1	Sq[t]A*	1	2	1	2	1	2
Sq[t]A[t2]*	1	2	1	2	1	2	Sq[t]G*	2	1	1	1	1	1	Sq[t]G[t2]*	2	1	1	1	1	1
Sq[t]L*	1	1	1	1	1	1	Sq[t]L[t2]*	2	1	1	1	1	1	Sq[t]S*	2	1	1	1	1	1
Sq[t]S[t2]*	2	1	1	1	1	1	SwA*	1	1	1	1	1	1	SwA[t]*	2	1	1	1	1	1
SwG*	1	1	1	2	1	2	SwG[t]*	1	1	1	1	1	2	SwL*	1	1	1	1	2	1
SwL[t]*	1	1	1	1	1	1	SwS*	1	1	2	1	2	1	SwS[t]*	1	1	1	1	2	1
Sw[t]A*	2	1	1	1	1	1	Sw[t]A[t2]*	2	1	1	1	1	1	Sw[t]G*	1	1	1	1	1	2
Sw[t]G[t2]*	1	1	1	1	1	2	Sw[t]L*	1	1	1	1	1	1	Sw[t]L[t2]*	1	1	1	1	1	1
Sw[t]S*	1	2	2	1	1	1	Sw[t]S[t2]*	1	1	2	1	1	1	SxA*	1	1	1	1	1	1
SxA[t]*	2	1	1	1	1	1	SxG*	1	1	1	1	1	1	SxG[t]*	1	1	1	1	1	1
SxL*	2	1	1	1	2	1	SxL[t]*	1	1	1	1	1	1	SxS*	2	1	2	1	2	1
SxS[t]*	1	1	1	1	2	1	Sx[t]A*	2	1	1	1	1	1	Sx[t]A[t2]*	2	1	1	1	1	2
Sx[t]G*	1	1	1	1	1	1	Sx[t]G[t2]*	1	1	1	2	1	1	Sx[t]L*	1	1	1	1	1	1
Sx[t]L[t2]*	1	1	1	1	1	1	Sx[t]S*	1	2	2	1	2	1	Sx[t]S[t2]*	1	1	1	1	2	1

Table C.5: Over all ranking of channel functions for each match cost function at 95% confidence for each data set in the HBPG framework.

Appendix D

Channel Norm Function Rankings For Each Global Stereopsis Framework

Noise→	None								Low Magnitude								High Magnitude							
Measure	**A**	**A[t]**	**G**	**G[t]**	**L**	**L[t]**	**S**	**S[t]**	**A**	**A[t]**	**G**	**G[t]**	**L**	**L[t]**	**S**	**S[t]**	**A**	**A[t]**	**G**	**G[t]**	**L**	**L[t]**	**S**	**S[t]**
Mx*B	2	1	2	1	3	1	4	3	1	1	1	1	2	2	2	2	1	1	1	2	3	3	4	3
Mx*D	2	2	2	1	2	2	4	3	1	1	1	2	2	2	2	1	1	2	1	2	2	2	2	2
S*B	2	1	1	1	3	1	4	3	1	1	1	2	4	3	4	3	1	1	1	1	2	2	2	2
S*D	2	1	1	1	2	1	4	3	1	1	2	1	1	1	1	1	2	3	2	1	1	2	1	1
S[t]*B	1	1	1	1	1	1	2	2	1	1	1	1	2	2	2	2	1	2	3	3	4	4	4	4
S[t]*D	1	1	1	1	1	2	2	2	2	2	1	1	1	1	1	1	2	2	1	1	1	1	1	1
Sq*B	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Sq*D	2	2	1	1	1	1	2	1	3	3	2	1	1	1	1	1	2	3	1	1	1	1	1	1
Sq[t]*B	2	2	1	1	1	1	1	1	2	2	1	1	1	1	1	1	2	2	1	1	1	1	1	1
Sq[t]*D	3	3	1	1	1	2	1	1	3	3	1	1	1	2	1	2	3	3	1	1	1	1	2	2
Sw*B	2	1	2	2	3	2	4	3	1	1	1	1	2	2	3	2	1	1	1	2	3	3	3	3
Sw*D	2	1	2	1	3	2	4	3	2	2	2	2	2	2	2	1	2	2	2	1	1	1	1	1
Sw[t]*B	1	1	1	2	2	2	3	3	1	2	2	2	3	3	3	3	1	1	1	2	3	3	3	3
Sw[t]*D	1	1	1	1	2	2	3	2	2	1	2	2	2	2	1	1	3	3	1	2	1	2	1	1
Sx*B	2	1	2	1	3	1	4	3	1	1	1	1	3	2	3	2	1	1	1	2	3	3	3	3
Sx*D	1	1	1	1	1	2	4	3	2	2	2	1	1	1	1	1	3	4	2	1	1	1	1	1
Sx[t]*B	1	1	1	1	1	2	2	2	1	1	2	1	3	3	3	3	1	1	2	2	3	3	3	3
Sx[t]*D	1	1	1	1	2	2	3	3	2	3	1	1	1	1	1	1	2	2	1	1	1	1	1	1

Table D.1: Over all ranking of channel norm functions for each match cost function at 95% confidence for each data set in the SO1 framework.

Noise→	None								Low Magnitude								High Magnitude							
Measure	**A**	**A[t]**	**G**	**G[t]**	**L**	**L[t]**	**S**	**S[t]**	**A**	**A[t]**	**G**	**G[t]**	**L**	**L[t]**	**S**	**S[t]**	**A**	**A[t]**	**G**	**G[t]**	**L**	**L[t]**	**S**	**S[t]**
Mx*B	1	1	2	1	3	2	4	3	1	1	2	1	2	2	3	2	1	1	2	2	2	2	2	2
Mx*D	2	1	1	1	2	3	5	4	1	1	1	1	1	1	1	1	1	3	1	2	2	2	2	1
S*B	2	1	2	2	3	2	4	3	1	1	2	2	3	2	3	2	1	1	2	1	2	2	2	2
S*D	2	1	1	1	1	2	4	3	1	1	2	1	1	1	1	1	2	3	2	1	2	2	1	1
S[t]*B	1	1	2	2	2	2	3	3	1	1	2	2	2	2	2	2	1	1	2	1	1	2	1	1
S[t]*D	1	1	1	1	2	2	2	2	2	2	1	1	1	1	1	1	3	4	1	2	2	2	1	1
Sq*B	1	1	1	1	1	1	1	1	1	1	1	1	1	2	1	1	1	2	1	1	1	1	1	1
Sq*D	1	1	1	1	1	1	1	1	2	3	1	1	1	1	1	1	2	3	1	1	1	1	1	1
Sq[t]*B	2	2	1	1	1	1	1	1	2	2	1	1	1	1	1	1	3	3	2	2	1	1	1	1
Sq[t]*D	2	2	1	1	1	1	1	1	3	3	1	1	1	1	2	2	3	4	1	1	1	1	2	2
Sw*B	2	1	2	1	3	2	4	3	1	1	1	2	3	2	3	2	1	1	2	1	2	2	2	2
Sw*D	2	1	1	1	2	3	5	4	1	2	2	1	1	1	1	1	2	3	1	1	1	1	1	1
Sw[t]*B	1	1	1	1	2	2	3	3	1	1	1	2	3	2	3	2	1	1	1	1	2	2	1	2
Sw[t]*D	1	1	1	1	2	2	3	3	2	2	1	1	1	2	1	1	3	4	1	1	2	2	1	2
Sx*B	1	1	2	1	3	2	4	3	1	1	2	2	3	2	4	3	1	1	3	3	3	2	3	3
Sx*D	1	1	1	1	1	2	4	3	3	2	3	1	2	1	1	1	2	3	1	1	1	1	1	1
Sx[t]*B	1	2	2	2	3	3	4	4	1	1	2	2	2	2	2	2	1	1	1	2	2	2	2	2
Sx[t]*D	1	1	1	1	2	2	2	2	2	2	1	1	1	1	1	1	3	3	1	1	1	2	1	1

Table D.2: Over all ranking of channel norm functions for each match cost function at 95% confidence for each data set in the SO2 framework.

Noise→	**None**								**Low Magnitude**								**High Magnitude**							
Measure	**A**	**A[t]**	**G**	**G[t]**	**L**	**L[t]**	**S**	**S[t]**	**A**	**A[t]**	**G**	**G[t]**	**L**	**L[t]**	**S**	**S[t]**	**A**	**A[t]**	**G**	**G[t]**	**L**	**L[t]**	**S**	**S[t]**
Mx*B	1	1	1	1	2	1	3	2	1	1	1	1	1	1	2	1	1	1	1	1	1	1	2	1
Mx*D	1	1	1	1	2	1	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
S*B	1	1	1	1	3	1	3	2	1	1	1	1	1	1	2	1	1	1	1	1	1	1	2	1
S*D	1	1	1	1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
S[t]*B	1	1	1	1	1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
S[t]*D	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Sq*B	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Sq*D	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Sq[t]*B	1	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Sq[t]*D	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Sw*B	1	1	1	1	2	1	3	2	1	1	1	1	2	1	2	1	1	1	1	1	1	1	1	1
Sw*D	1	1	1	1	2	1	2	1	1	1	1	1	2	1	2	1	1	1	1	1	1	1	1	1
Sw[t]*B	1	1	1	1	1	1	2	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Sw[t]*D	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Sx*B	2	1	1	1	3	1	4	3	1	1	1	1	1	1	2	1	2	1	1	1	1	1	2	2
Sx*D	1	1	1	1	1	1	2	1	1	1	1	1	1	1	2	1	1	1	1	1	1	1	1	1
Sx[t]*B	1	1	1	1	1	1	2	3	1	1	1	1	1	1	1	1	1	1	1	1	1	1	2	2
Sx[t]*D	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Table D.3: Over all ranking of channel norm functions for each match cost function at 95% confidence for each data set in the RTree framework.

Noise→	None								Low Magnitude								High Magnitude							
Measure	**A**	**A[t]**	**G**	**G[t]**	**L**	**L[t]**	**S**	**S[t]**	**A**	**A[t]**	**G**	**G[t]**	**L**	**L[t]**	**S**	**S[t]**	**A**	**A[t]**	**G**	**G[t]**	**L**	**L[t]**	**S**	**S[t]**
Mx*B	3	1	1	1	3	2	3	3	2	1	1	1	2	2	3	2	3	1	1	2	3	3	4	3
Mx*D	3	1	2	1	3	2	4	3	1	1	2	1	2	2	3	1	1	1	1	1	2	2	3	2
S*B	2	1	1	1	2	1	2	2	3	1	1	1	3	2	4	3	3	1	1	1	4	2	5	4
S*D	3	1	2	1	4	1	5	4	1	1	2	1	2	2	3	1	1	1	1	1	1	1	1	1
S[t]*B	1	1	1	1	1	1	2	2	1	1	1	2	1	2	3	3	1	1	1	1	1	2	3	3
S[t]*D	1	1	1	1	2	2	2	2	1	1	1	1	1	1	1	1	2	2	1	1	1	1	1	1
Sq*B	1	1	1	1	3	2	2	2	1	1	1	1	3	2	3	3	1	1	1	1	3	2	3	3
Sq*D	2	1	2	1	2	2	3	2	1	1	2	1	2	1	1	1	1	1	1	1	1	1	1	1
Sq[t]*B	2	1	1	1	1	2	1	1	1	1	1	1	1	2	1	1	1	1	1	1	1	2	1	1
Sq[t]*D	2	2	1	1	1	1	1	1	2	2	1	1	1	1	1	1	2	2	1	1	1	1	1	1
Sw*B	3	2	3	1	4	2	4	3	2	1	2	2	3	2	3	3	2	1	2	2	4	2	3	3
Sw*D	2	1	2	1	2	2	4	3	1	1	1	1	1	1	1	1	1	1	3	1	1	2	1	1
Sw[t]*B	1	1	1	1	1	2	2	2	1	1	1	2	1	1	2	3	1	1	1	1	1	1	2	2
Sw[t]*D	1	1	2	1	3	2	4	4	1	1	2	1	2	2	2	1	1	1	2	1	1	2	1	1
Sx*B	2	1	1	1	2	1	2	1	2	1	1	1	2	2	3	2	2	1	1	1	3	2	3	3
Sx*D	4	1	3	2	3	3	6	5	1	1	2	1	1	1	2	1	1	1	1	1	1	1	1	1
Sx[t]*B	1	1	1	1	1	1	2	2	1	1	1	1	1	2	2	2	1	1	2	2	2	2	2	3
Sx[t]*D	1	1	1	1	2	2	3	3	1	1	2	2	1	2	1	1	2	2	1	1	1	1	1	1

Table D.4: Over all ranking of channel norm functions for each match cost function at 95% confidence for each data set in the HBPTL framework.

Noise→	None								Low Magnitude								High Magnitude							
Measure	**A**	**A[t]**	**G**	**G[t]**	**L**	**L[t]**	**S**	**S[t]**	**A**	**A[t]**	**G**	**G[t]**	**L**	**L[t]**	**S**	**S[t]**	**A**	**A[t]**	**G**	**G[t]**	**L**	**L[t]**	**S**	**S[t]**
Mx*B	3	2	2	1	4	2	5	2	1	1	1	1	2	1	2	2	2	1	1	1	3	1	3	3
Mx*D	3	1	2	1	3	2	5	4	1	1	1	1	2	1	3	2	2	1	1	1	2	1	3	2
S*B	3	2	2	1	4	2	5	4	2	1	1	1	3	1	3	3	2	1	1	1	3	2	4	4
S*D	2	1	1	1	2	1	4	3	1	1	2	1	2	1	2	1	1	1	2	1	1	1	2	1
S[t]*B	1	1	1	1	1	1	2	2	1	1	1	1	1	1	2	2	1	1	1	1	1	1	2	2
S[t]*D	1	1	1	1	1	1	2	2	1	1	1	1	1	1	1	1	1	2	1	1	1	1	1	1
Sq*B	2	2	2	1	3	2	3	3	1	1	1	1	1	1	2	1	1	1	1	1	2	1	2	2
Sq*D	2	1	1	1	2	1	3	2	1	1	2	1	2	1	2	1	1	1	1	1	1	1	1	1
Sq[t]*B	1	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Sq[t]*D	2	3	1	1	1	1	1	1	2	2	1	1	1	1	1	1	2	2	1	1	1	1	1	1
Sw*B	1	1	1	1	3	1	4	2	1	1	1	1	2	1	2	2	2	1	1	1	3	1	4	3
Sw*D	3	1	3	2	4	3	5	4	1	1	2	1	2	1	1	1	1	1	1	1	1	1	1	1
Sw[t]*B	2	1	1	1	1	2	3	3	1	1	1	1	1	1	2	2	1	1	1	1	1	1	2	2
Sw[t]*D	1	1	1	1	2	1	3	3	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Sx*B	1	1	1	1	3	1	3	2	1	1	1	1	2	1	3	2	1	1	1	1	2	1	2	2
Sx*D	2	1	1	1	2	1	4	3	1	1	1	1	1	1	1	1	2	2	2	1	1	1	1	1
Sx[t]*B	1	1	1	1	1	1	2	2	1	1	1	1	1	1	3	2	1	1	1	1	1	1	2	1
Sx[t]*D	1	1	1	1	1	1	2	2	1	1	1	2	1	1	1	1	2	3	1	1	1	1	1	1

Table D.5: Over all ranking of channel norm functions for each match cost function at 95% confidence for each data set in the HBPG framework.

Appendix E

Channel Aggregate Rankings For Each Global Stereopsis Framework

Measure	**Mx**	**S**	**S[t]**	**Sq**	**Sq[t]**	**Sw**	**Sw[t]**	**Sx**	**Sx[t]**
*AB	1	2	1	1	3	1	1	3	3
*AD	1	2	1	2	4	2	1	3	3
*A[t]B	1	1	1	2	3	1	1	3	3
*A[t]D	1	2	2	2	4	2	2	3	3
*GB	1	1	1	1	1	1	1	3	2
*GD	1	3	2	1	1	3	2	4	4
*G[t]B	1	1	1	1	1	2	2	3	3
*G[t]D	1	2	2	2	2	2	3	4	4
*LB	2	2	1	1	1	3	1	5	4
*LD	1	2	2	1	1	3	2	4	4
*L[t]B	1	1	1	1	1	1	2	3	3
*L[t]D	1	1	2	1	1	2	2	3	3
*SB	3	3	2	1	1	3	2	5	4
*SD	3	4	2	2	1	4	3	6	5
*S[t]B	3	3	3	2	1	3	3	4	4
*S[t]D	2	3	2	1	1	3	3	5	4

(a) Rankings for data set with no noise introduced.

Measure	**Mx**	**S**	**S[t]**	**Sq**	**Sq[t]**	**Sw**	**Sw[t]**	**Sx**	**Sx[t]**
*AB	1	2	2	1	4	1	1	3	3
*AD	1	2	5	3	6	1	1	4	4
*A[t]B	1	1	2	1	4	1	1	3	3
*A[t]D	1	2	5	3	6	1	1	4	5
*GB	1	1	2	1	1	1	1	3	3
*GD	1	1	1	1	1	1	1	3	2
*G[t]B	1	2	2	1	1	1	1	3	3
*G[t]D	1	1	1	1	1	1	1	2	2
*LB	2	3	2	1	1	3	2	5	4
*LD	1	2	1	1	1	1	1	3	3
*L[t]B	2	2	2	1	1	2	2	3	3
*L[t]D	1	1	1	1	1	1	1	2	2
*SB	3	3	2	1	1	3	2	5	4
*SD	2	2	2	1	2	1	1	3	3
*S[t]B	2	2	2	1	1	2	2	3	3
*S[t]D	2	2	2	2	2	1	1	3	3

(b) Rankings for data set with lowest magnitude noise introduced.

Measure	**Mx**	**S**	**S[t]**	**Sq**	**Sq[t]**	**Sw**	**Sw[t]**	**Sx**	**Sx[t]**
*AB	2	3	3	1	5	1	1	4	4
*AD	2	3	7	4	8	1	1	5	6
*A[t]B	2	3	4	2	6	1	1	5	5
*A[t]D	2	2	6	3	7	1	1	4	5
*GB	1	3	4	1	2	1	1	5	5
*GD	3	4	4	3	3	2	1	5	4
*G[t]B	2	3	4	1	2	1	1	5	6
*G[t]D	4	4	4	3	3	1	2	4	4
*LB	3	4	4	1	1	2	2	5	5
*LD	2	2	3	2	2	1	1	4	4
*L[t]B	3	3	3	1	1	2	2	5	4
*L[t]D	2	2	2	2	2	1	1	3	3
*SB	3	3	3	1	1	2	1	4	4
*SD	3	2	3	2	4	1	1	3	3
*S[t]B	3	4	4	1	1	2	2	5	5
*S[t]D	4	4	4	3	6	1	2	4	5

(c) Rankings for data set with highest magnitude noise introduced.

Table E.1: Ranking of channel aggregates for each match cost function at 95% confidence in the SO1 framework.

Measure	Mx	S	S[t]	Sq	Sq[t]	Sw	Sw[t]	Sx	Sx[t]
*AB	1	1	1	1	2	2	1	4	3
*AD	1	2	1	1	3	1	1	2	2
*A[t]B	1	1	1	1	2	1	1	2	2
*A[t]D	1	2	2	2	4	2	2	3	3
*GB	1	2	1	1	1	2	1	3	3
*GD	1	1	1	1	1	1	1	2	2
*G[t]B	1	1	1	1	1	1	1	2	2
*G[t]D	1	1	2	1	1	2	1	3	3
*LB	3	3	1	1	1	3	2	5	4
*LD	1	1	3	1	1	2	3	3	4
*L[t]B	2	2	2	1	1	2	2	3	3
*L[t]D	3	3	4	2	1	4	4	5	5
*SB	4	4	2	1	1	3	2	5	4
*SD	5	5	3	2	1	5	4	6	5
*S[t]B	3	3	3	2	1	3	3	4	4
*S[t]D	2	3	2	1	1	3	3	5	4

(a) Rankings for data set with no noise introduced.

Measure	Mx	S	S[t]	Sq	Sq[t]	Sw	Sw[t]	Sx	Sx[t]
*AB	1	1	1	1	3	1	1	2	2
*AD	1	1	5	2	6	1	1	3	4
*A[t]B	1	1	1	1	3	1	1	2	2
*A[t]D	1	1	4	2	5	1	1	2	3
*GB	1	2	1	1	1	1	1	4	3
*GD	1	2	1	2	1	2	1	4	3
*G[t]B	1	1	2	1	1	2	2	3	3
*G[t]D	1	1	1	1	1	1	1	2	2
*LB	2	2	2	1	1	2	2	4	3
*LD	2	2	2	2	2	2	1	3	3
*L[t]B	1	2	1	1	1	1	1	3	3
*L[t]D	1	1	1	1	1	1	1	3	2
*SB	3	4	2	1	1	3	3	6	5
*SD	2	1	1	1	3	1	1	2	2
*S[t]B	2	2	2	1	1	2	2	3	3
*S[t]D	2	2	1	1	3	1	1	3	3

(b) Rankings for data set with lowest magnitude noise introduced.

Measure	Mx	S	S[t]	Sq	Sq[t]	Sw	Sw[t]	Sx	Sx[t]
*AB	1	2	2	1	4	1	1	3	3
*AD	1	2	5	3	5	1	2	4	5
*A[t]B	2	2	3	3	6	1	1	4	5
*A[t]D	4	3	6	5	6	1	2	4	6
*GB	2	3	3	1	2	1	1	5	4
*GD	3	3	3	3	4	2	1	4	3
*G[t]B	2	2	2	2	2	1	1	3	3
*G[t]D	2	2	2	2	3	1	1	3	3
*LB	2	2	2	1	1	1	1	3	3
*LD	3	2	2	2	3	1	1	3	3
*L[t]B	2	2	2	1	1	1	1	3	3
*L[t]D	3	2	2	2	3	1	1	3	3
*SB	2	2	2	1	1	1	1	4	3
*SD	3	2	2	2	4	1	1	3	3
*S[t]B	2	2	2	1	1	1	1	3	3
*S[t]D	3	3	3	3	5	1	2	4	4

(c) Rankings for data set with highest magnitude noise introduced.

Table E.2: Ranking of channel aggregates for each match cost function at 95% confidence in the SO2 framework.

Measure	Mx	S	S[t]	Sq	Sq[t]	Sw	Sw[t]	Sx	Sx[t]
*AB	1	1	1	1	1	1	1	1	1
*AD	1	1	1	1	1	1	2	1	1
*A[t]B	1	1	1	1	2	1	1	1	1
*A[t]D	1	1	1	1	2	1	1	1	1
*GB	1	1	1	1	1	1	1	1	1
*GD	1	1	1	1	1	1	1	1	1
*G[t]B	1	1	1	1	1	1	1	1	1
*G[t]D	1	1	1	1	1	1	1	1	1
*LB	2	3	1	1	1	3	2	3	2
*LD	2	2	1	1	1	2	1	1	1
*L[t]B	1	1	1	1	1	1	1	1	1
*L[t]D	1	1	1	1	1	1	1	1	1
*SB	2	2	1	1	1	2	1	2	1
*SD	1	2	1	1	1	2	1	2	1
*S[t]B	3	2	2	2	1	3	3	3	3
*S[t]D	1	1	1	1	1	1	1	1	1

(a) Rankings for data set with no noise introduced.

Measure	Mx	S	S[t]	Sq	Sq[t]	Sw	Sw[t]	Sx	Sx[t]
*AB	1	1	1	1	1	1	1	1	1
*AD	1	1	1	1	1	1	1	1	1
*A[t]B	1	1	1	1	1	1	1	1	1
*A[t]D	1	1	1	1	1	1	1	1	1
*GB	1	1	1	1	1	1	1	1	1
*GD	1	1	1	1	1	1	1	1	1
*G[t]B	1	1	1	1	1	1	1	1	1
*G[t]D	1	1	1	1	1	1	1	1	1
*LB	1	1	1	1	1	1	1	1	1
*LD	1	1	1	1	1	2	1	1	1
*L[t]B	1	1	1	1	1	1	1	1	1
*L[t]D	1	1	1	1	1	1	1	1	1
*SB	2	2	1	1	1	1	1	2	1
*SD	1	1	1	1	1	2	1	2	1
*S[t]B	1	1	1	1	1	1	1	1	1
*S[t]D	1	1	1	1	1	1	1	1	1

(b) Rankings for data set with lowest magnitude noise introduced.

Measure	Mx	S	S[t]	Sq	Sq[t]	Sw	Sw[t]	Sx	Sx[t]
*AB	1	1	1	1	1	1	1	2	1
*AD	1	1	1	1	1	1	1	1	1
*A[t]B	1	1	1	1	1	1	1	1	1
*A[t]D	1	1	1	1	2	1	1	1	1
*GB	1	1	1	1	1	1	1	1	1
*GD	1	1	1	1	1	1	1	1	1
*G[t]B	1	1	1	1	1	1	1	1	1
*G[t]D	1	1	1	1	1	1	1	1	1
*LB	1	1	1	1	1	1	1	1	1
*LD	1	1	1	1	1	1	1	1	1
*L[t]B	1	1	1	1	1	1	1	1	1
*L[t]D	1	1	1	1	1	1	1	1	1
*SB	1	2	1	1	1	1	1	2	1
*SD	1	1	1	1	1	1	1	2	1
*S[t]B	1	1	1	1	1	1	1	1	1
*S[t]D	1	1	1	1	1	1	1	1	1

(c) Rankings for data set with highest magnitude noise introduced.

Table E.3: Ranking of channel aggregates for each match cost function at 95% confidence in the RTree framework.

Measure	Mx	S	S[t]	Sq	Sq[t]	Sw	Sw[t]	Sx	Sx[t]
*AB	3	3	1	1	1	2	1	4	3
*AD	2	3	1	2	5	3	1	4	2
*A[t]B	1	1	1	1	1	1	2	3	3
*A[t]D	1	1	1	1	3	2	1	2	2
*GB	1	1	1	1	1	2	1	3	2
*GD	1	2	1	2	1	3	1	3	2
*G[t]B	1	1	1	1	1	1	1	2	2
*G[t]D	1	1	1	1	1	1	1	2	2
*LB	2	2	1	2	1	2	1	3	2
*LD	2	3	2	2	1	3	2	3	3
*L[t]B	1	1	1	1	1	1	1	3	2
*L[t]D	1	1	1	1	1	2	1	2	2
*SB	3	3	2	2	1	3	2	4	3
*SD	4	4	2	2	1	4	3	5	4
*S[t]B	2	2	2	2	1	2	2	3	3
*S[t]D	4	5	3	2	1	5	5	6	5

(a) Rankings for data set with no noise introduced.

Measure	Mx	S	S[t]	Sq	Sq[t]	Sw	Sw[t]	Sx	Sx[t]
*AB	3	3	1	1	1	3	2	4	3
*AD	1	1	1	1	4	2	1	3	2
*A[t]B	1	2	1	1	1	2	2	3	3
*A[t]D	1	1	2	1	4	2	2	3	3
*GB	1	2	1	1	1	2	1	3	3
*GD	1	3	2	2	1	3	2	4	3
*G[t]B	1	1	1	1	1	2	2	2	3
*G[t]D	1	1	1	1	1	2	2	3	4
*LB	3	3	1	2	1	3	2	4	3
*LD	2	2	2	2	1	2	2	3	3
*L[t]B	1	1	1	1	1	1	1	2	2
*L[t]D	2	2	1	1	1	2	2	2	3
*SB	4	4	2	2	1	3	2	5	4
*SD	3	3	2	2	1	3	2	4	3
*S[t]B	2	3	2	2	1	2	2	4	4
*S[t]D	1	1	1	1	1	2	2	3	2

(b) Rankings for data set with lowest magnitude noise introduced.

Measure	Mx	S	S[t]	Sq	Sq[t]	Sw	Sw[t]	Sx	Sx[t]
*AB	2	2	1	1	1	2	1	3	2
*AD	1	1	2	1	4	1	1	2	3
*A[t]B	1	1	1	1	1	1	1	2	2
*A[t]D	1	2	3	1	5	1	1	3	4
*GB	1	2	1	1	1	2	1	3	4
*GD	1	2	1	1	1	2	2	3	2
*G[t]B	1	2	1	1	1	2	2	3	3
*G[t]D	1	1	1	1	1	1	1	2	2
*LB	3	3	1	2	1	3	2	4	3
*LD	1	1	1	1	1	1	1	2	2
*L[t]B	1	1	1	1	1	2	1	3	3
*L[t]D	1	1	1	1	1	1	1	2	2
*SB	4	4	3	2	1	2	2	5	4
*SD	2	2	1	1	1	1	1	2	2
*S[t]B	3	4	3	2	1	2	3	5	5
*S[t]D	1	1	1	1	1	1	1	2	2

(c) Rankings for data set with highest magnitude noise introduced.

Table E.4: Ranking of channel aggregates for each match cost function at 95% confidence in the HBPTL framework.

Measure	Mx	S	S[t]	Sq	Sq[t]	Sw	Sw[t]	Sx	Sx[t]
*AB	2	2	1	2	1	2	2	3	3
*AD	2	2	1	2	2	2	1	3	2
*A[t]B	1	1	1	1	1	1	1	2	2
*A[t]D	1	2	1	2	4	2	2	3	3
*GB	2	2	1	1	1	2	1	3	3
*GD	2	2	1	2	1	3	2	4	2
*G[t]B	1	1	1	1	1	2	1	3	3
*G[t]D	1	2	1	2	1	2	2	4	3
*LB	2	3	1	2	1	3	1	4	2
*LD	2	4	1	3	1	4	2	5	2
*L[t]B	1	1	1	2	1	1	1	3	3
*L[t]D	1	1	1	1	1	2	1	3	3
*SB	3	3	2	2	1	4	2	5	3
*SD	3	4	2	2	1	4	3	5	4
*S[t]B	1	2	2	2	1	2	2	3	3
*S[t]D	3	4	2	2	1	4	5	6	5

(a) Rankings for data set with no noise introduced.

Measure	Mx	S	S[t]	Sq	Sq[t]	Sw	Sw[t]	Sx	Sx[t]
*AB	1	2	1	1	1	1	1	3	2
*AD	1	2	1	1	4	1	1	3	3
*A[t]B	1	1	1	1	1	1	1	2	2
*A[t]D	1	1	1	1	4	2	1	3	3
*GB	1	1	1	1	1	1	1	2	2
*GD	1	2	1	2	1	3	2	4	3
*G[t]B	1	1	1	1	1	1	1	2	2
*G[t]D	1	2	1	1	1	2	2	3	3
*LB	3	3	1	2	1	3	2	4	3
*LD	2	3	2	2	1	3	2	4	4
*L[t]B	1	1	1	1	1	1	1	2	2
*L[t]D	1	2	1	1	1	2	2	3	3
*SB	4	4	3	2	1	4	3	5	4
*SD	3	3	2	3	1	3	3	4	4
*S[t]B	2	2	2	1	1	2	2	3	2
*S[t]D	2	2	2	2	1	3	2	4	4

(b) Rankings for data set with lowest magnitude noise introduced.

Measure	Mx	S	S[t]	Sq	Sq[t]	Sw	Sw[t]	Sx	Sx[t]
*AB	2	3	1	1	1	2	1	4	4
*AD	1	1	2	1	4	1	1	3	3
*A[t]B	1	1	1	1	1	1	1	2	2
*A[t]D	1	1	3	2	5	1	2	3	4
*GB	1	1	1	1	1	1	1	2	2
*GD	1	2	2	1	1	2	2	3	3
*G[t]B	1	1	1	1	1	1	1	2	2
*G[t]D	1	2	1	1	1	2	2	2	3
*LB	3	3	1	2	1	3	1	4	3
*LD	1	1	1	1	1	1	1	2	2
*L[t]B	1	2	1	1	1	1	1	3	3
*L[t]D	1	1	1	1	1	1	1	2	2
*SB	4	4	4	2	1	4	3	5	4
*SD	2	2	1	1	1	1	1	2	2
*S[t]B	4	4	4	2	1	3	3	4	4
*S[t]D	2	2	2	1	1	2	2	2	3

(c) Rankings for data set with highest magnitude noise introduced.

Table E.5: Ranking of channel aggregates for each match cost function at 95% confidence in the HBPG framework.

Appendix F

Parameter Plots For The Weighted Sum Channel Aggregate

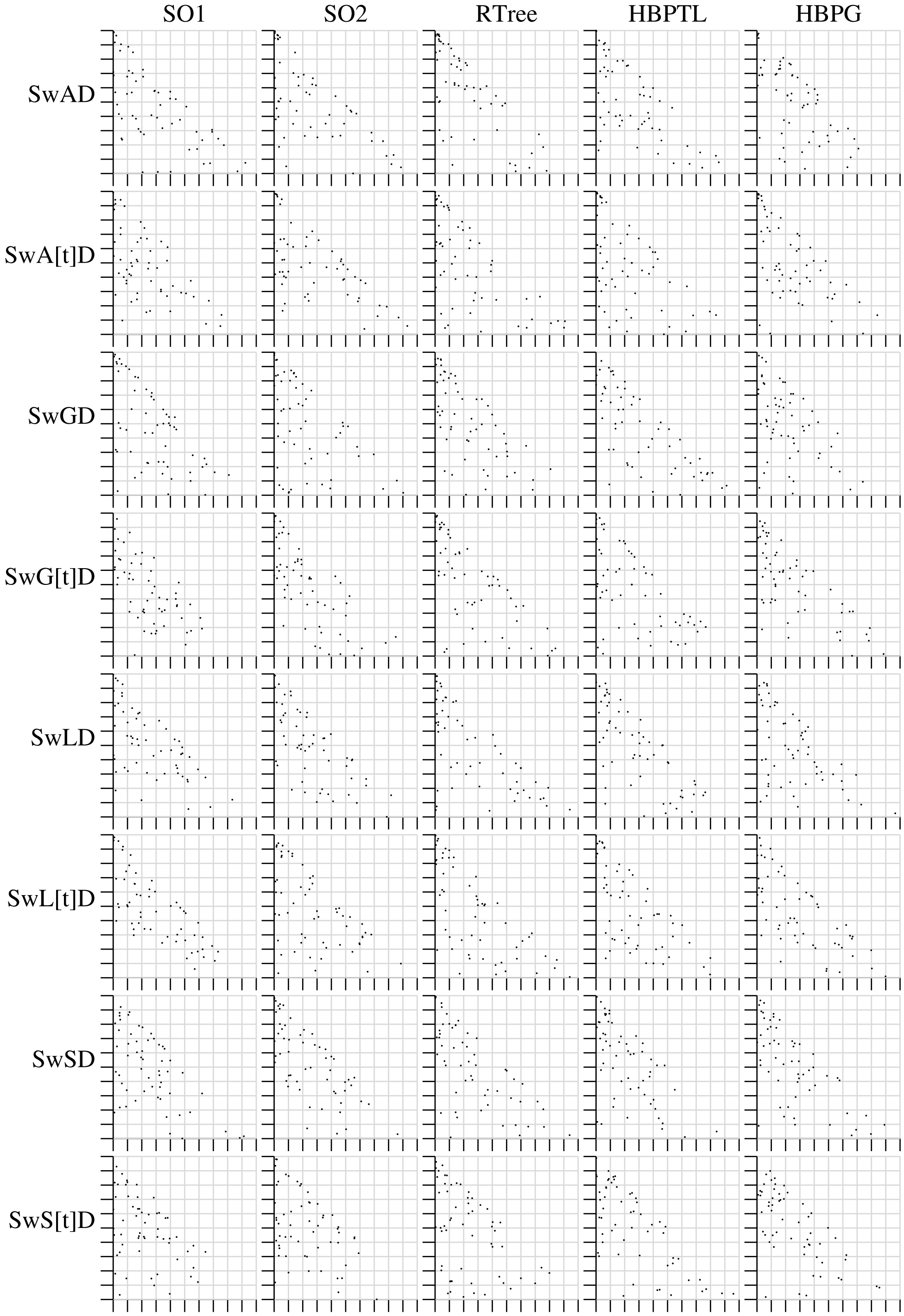


Figure F.1: Scatter plots of the blue and green weights of the weighted sum channel aggregate for all Sw*D form match cost functions on the data set with no noise.

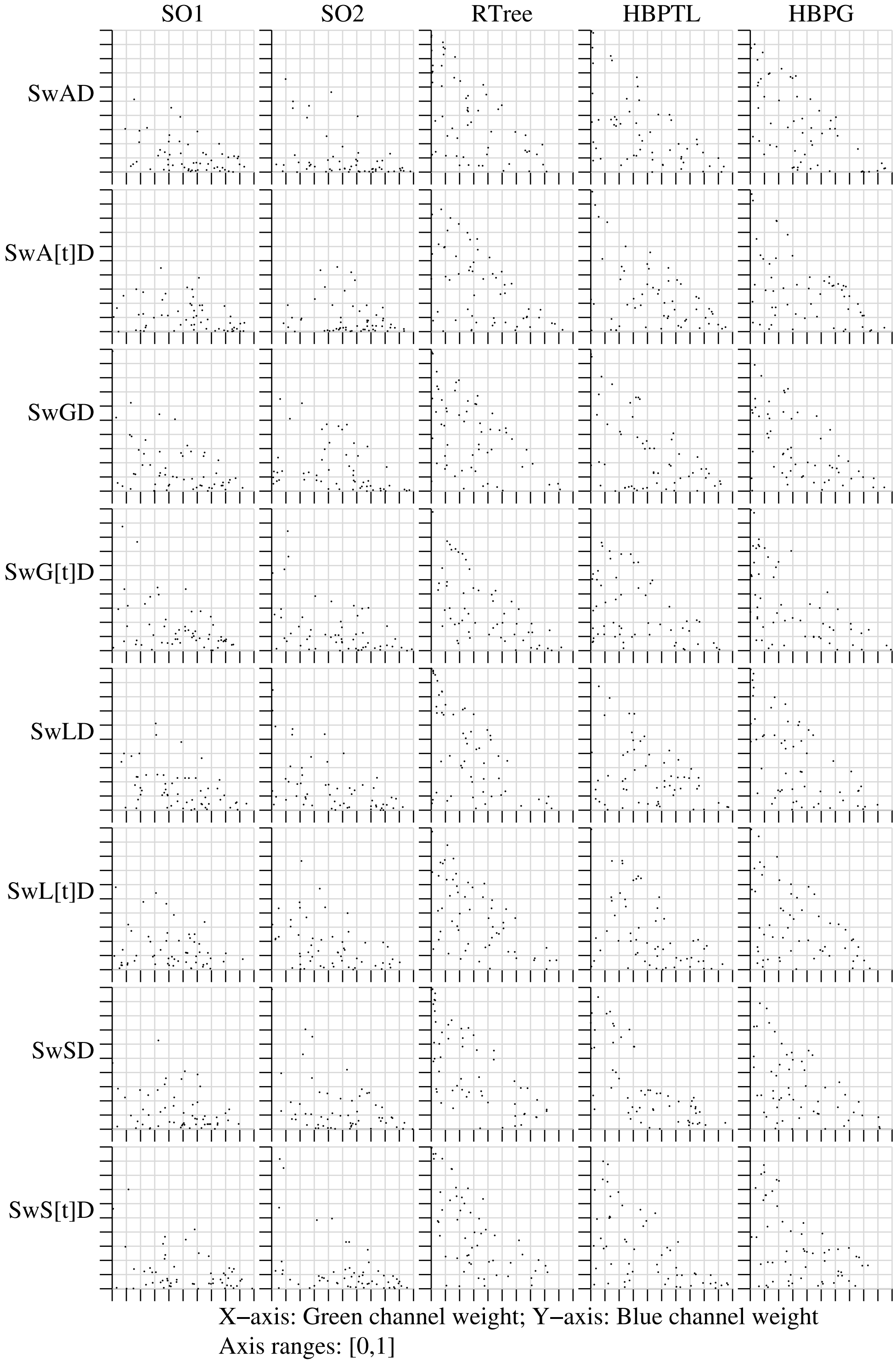


Figure F.2: Scatter plots of the blue and green weights of the weighted sum channel aggregate for all Sw*D form match cost functions on the data set with low magnitude noise.

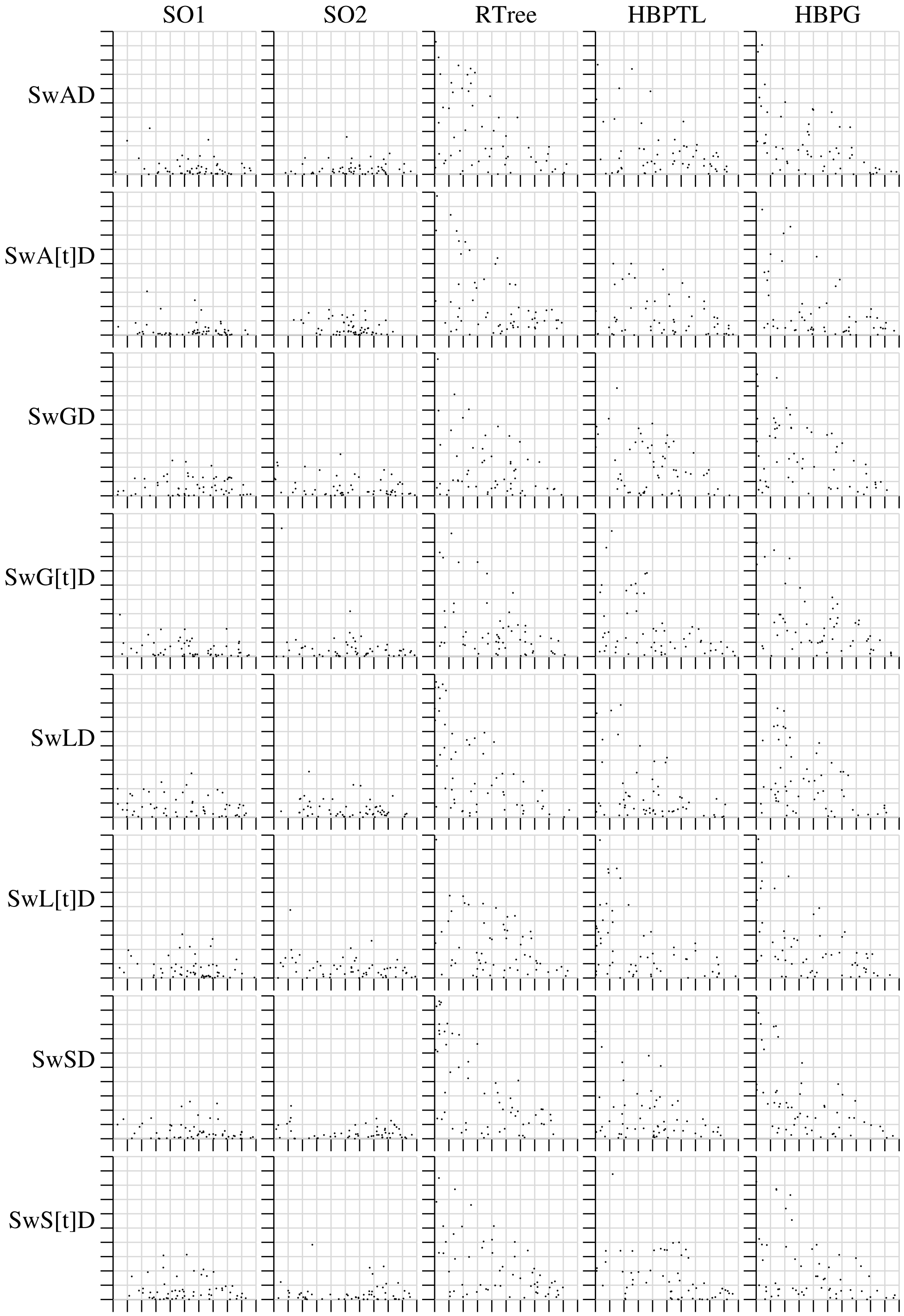


Figure F.3: Scatter plots of the blue and green weights of the weighted sum channel aggregate for all Sw*D form match cost functions on the data set with high magnitude noise.

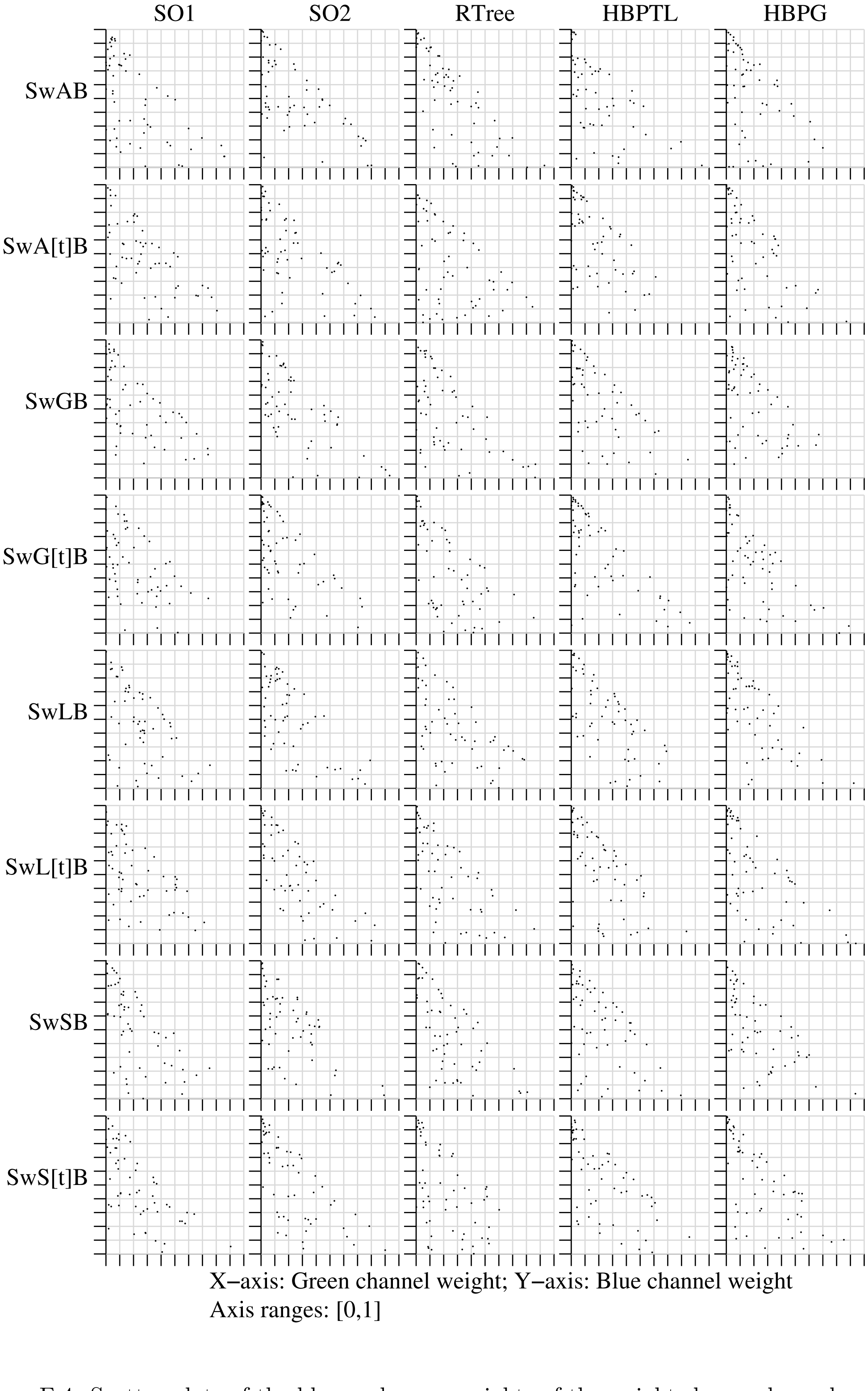


Figure F.4: Scatter plots of the blue and green weights of the weighted sum channel aggregate for all Sw*B form match cost functions on the data set with no noise.

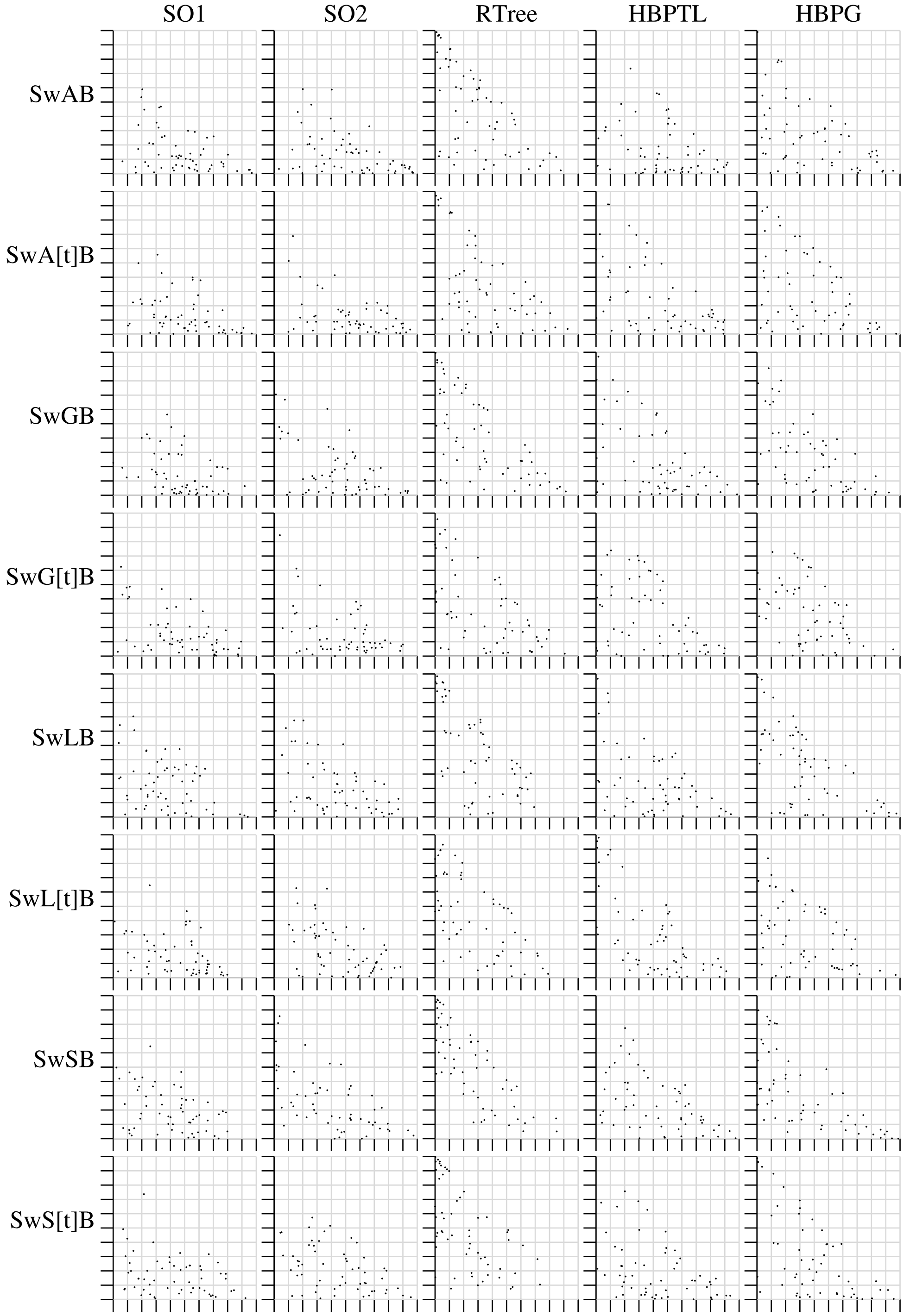


Figure F.5: Scatter plots of the blue and green weights of the weighted sum channel aggregate for all Sw*B form match cost functions on the data set with low magnitude noise.

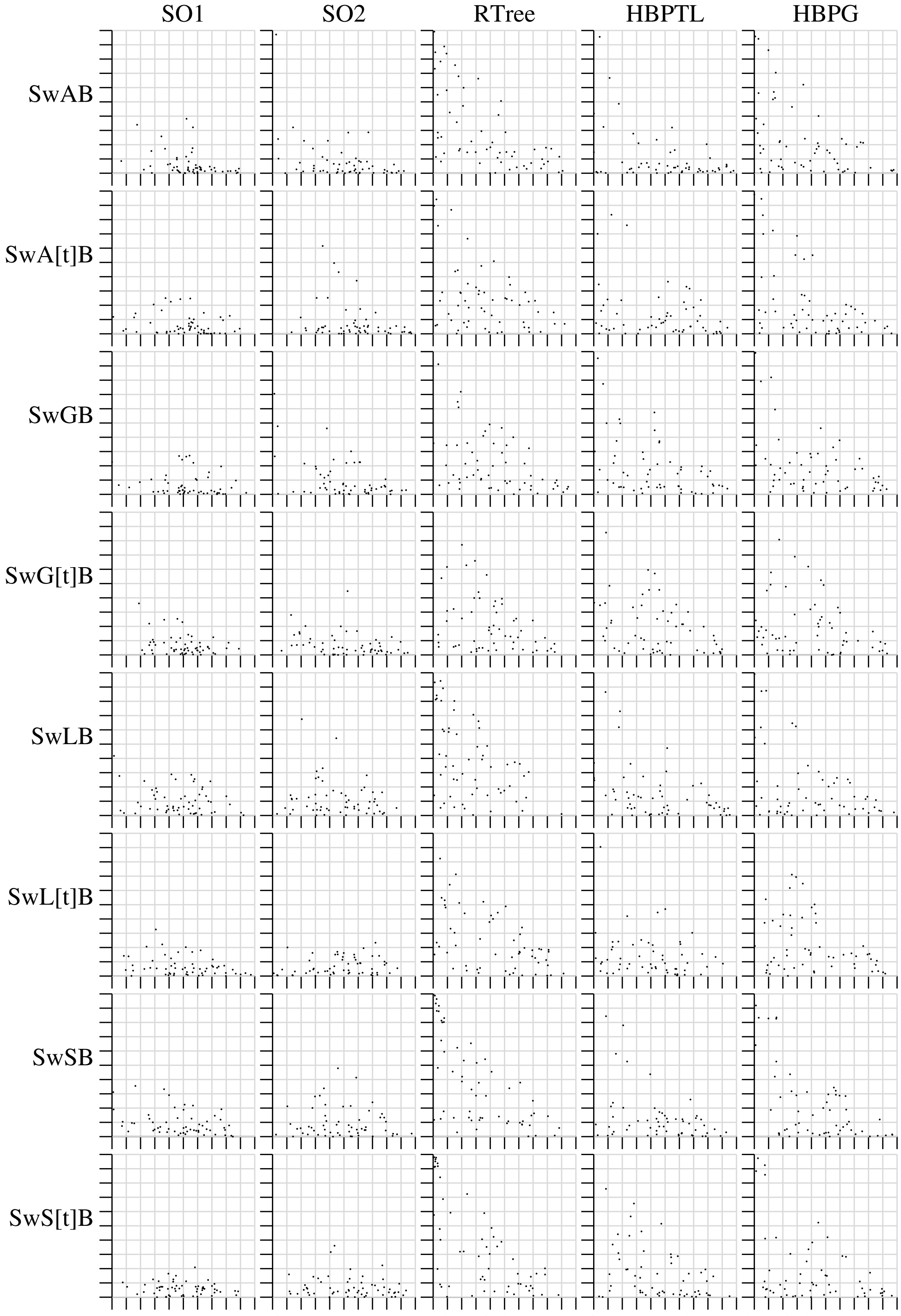


Figure F.6: Scatter plots of the blue and green weights of the weighted sum channel aggregate for all Sw*B form match cost functions on the data set with high magnitude noise.

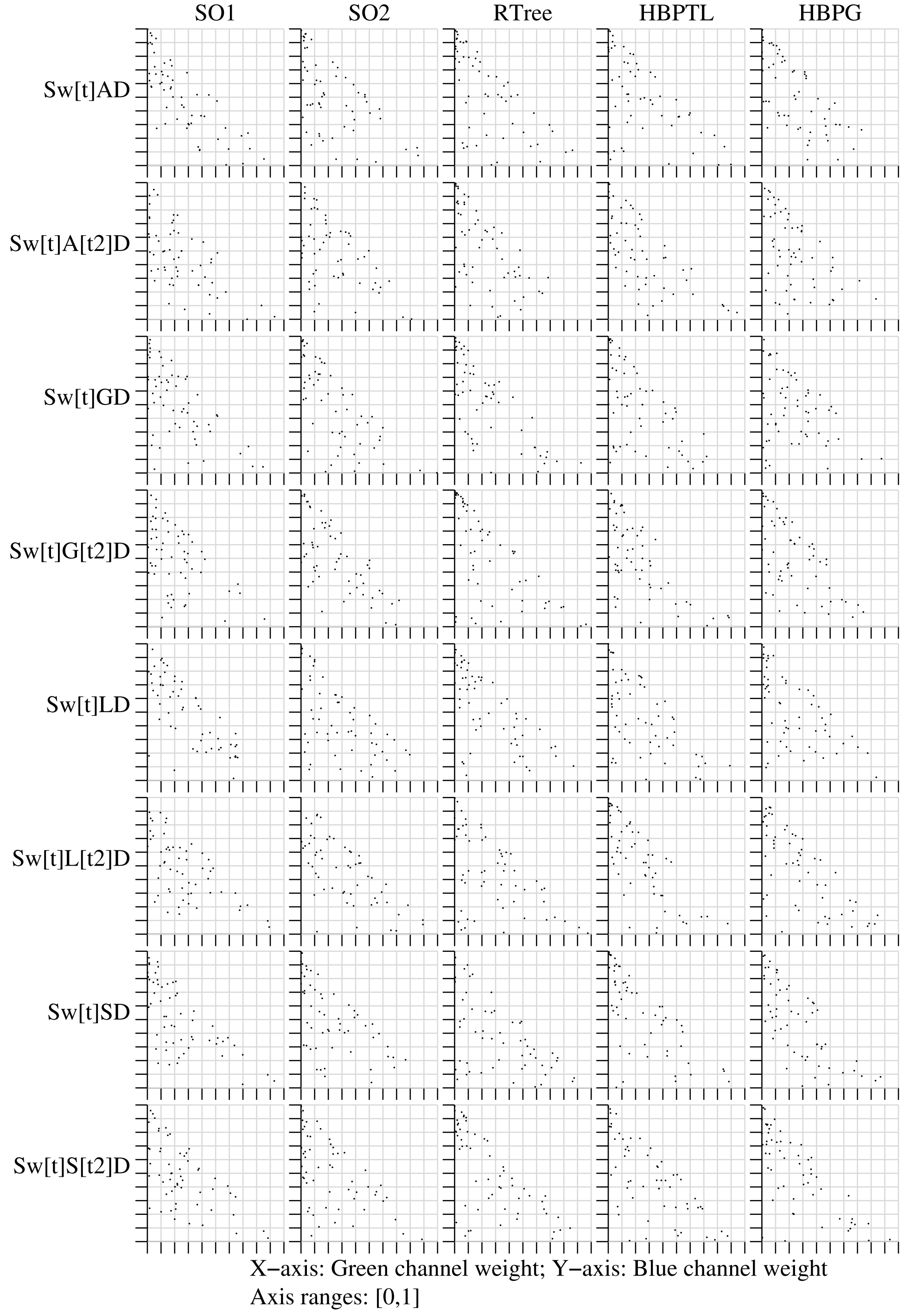


Figure F.7: Scatter plots of the blue and green weights of the weighted sum channel aggregate for all Sw[t]*D form match cost functions on the data set with no noise.

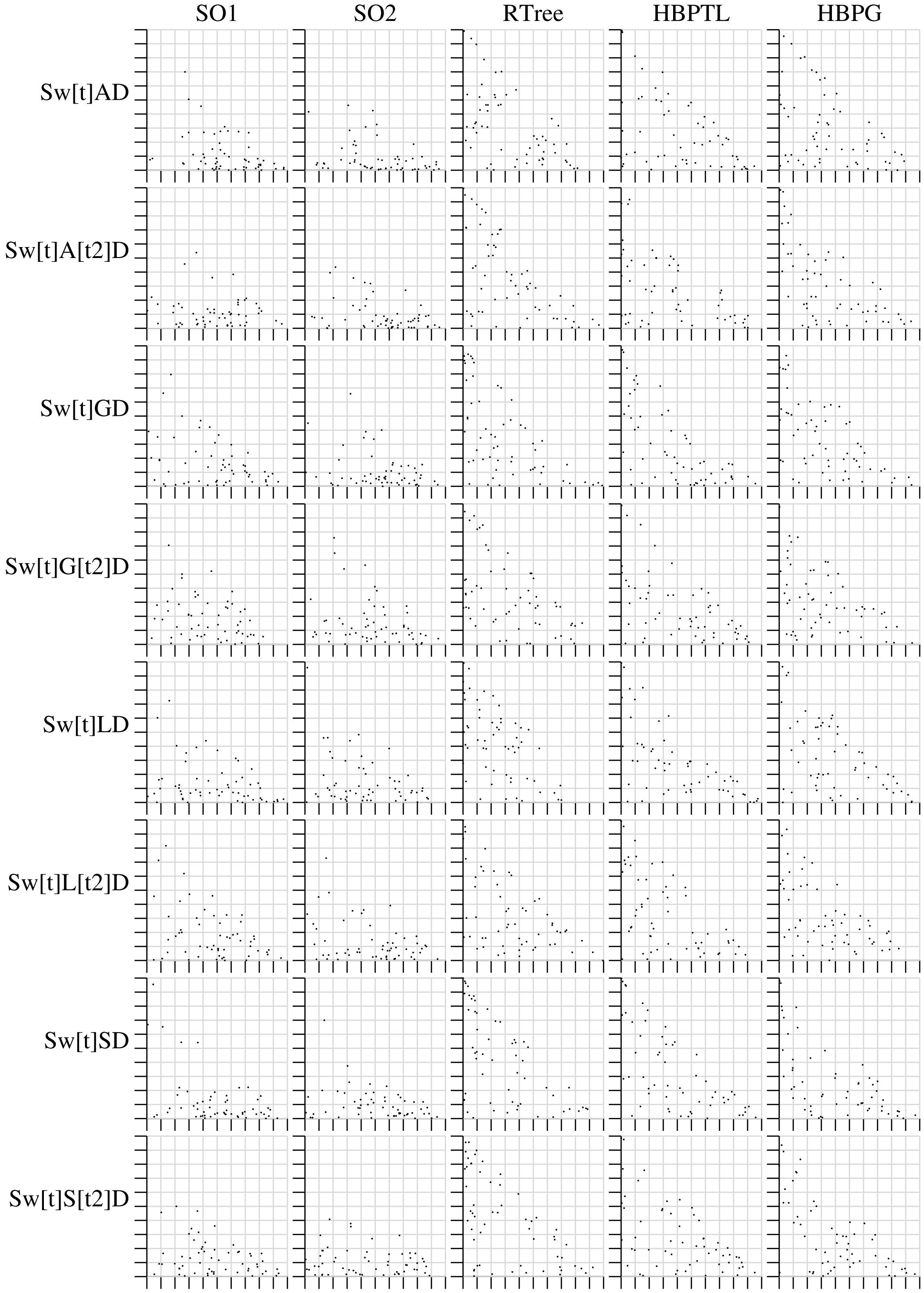


Figure F.8: Scatter plots of the blue and green weights of the weighted sum channel aggregate for all Sw[t]*D form match cost functions on the data set with low magnitude noise.

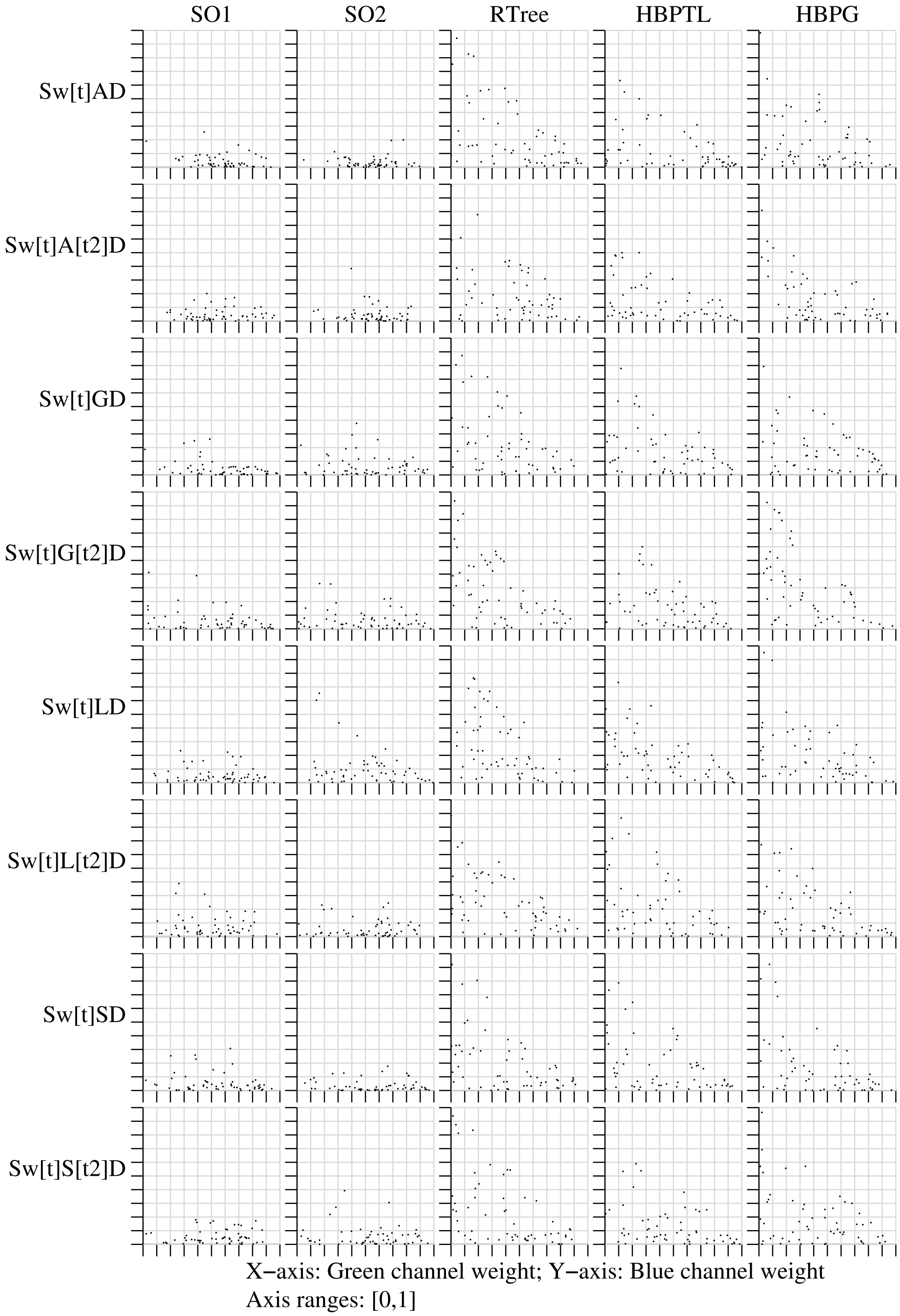


Figure F.9: Scatter plots of the blue and green weights of the weighted sum channel aggregate for all Sw[t]*D form match cost functions on the data set with high magnitude noise.

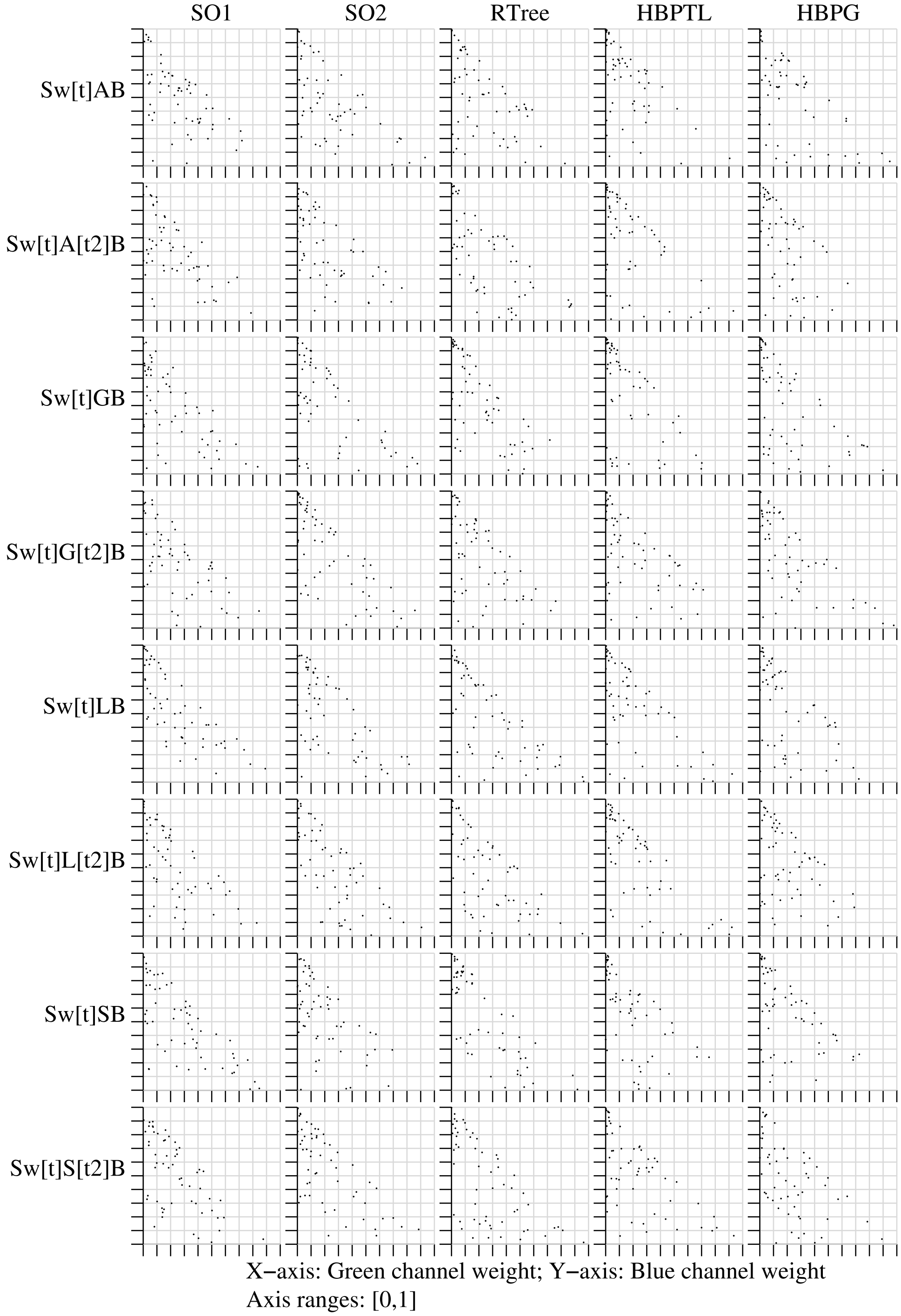


Figure F.10: Scatter plots of the blue and green weights of the weighted sum channel aggregate for all Sw[t]*B form match cost functions on the data set with no noise.

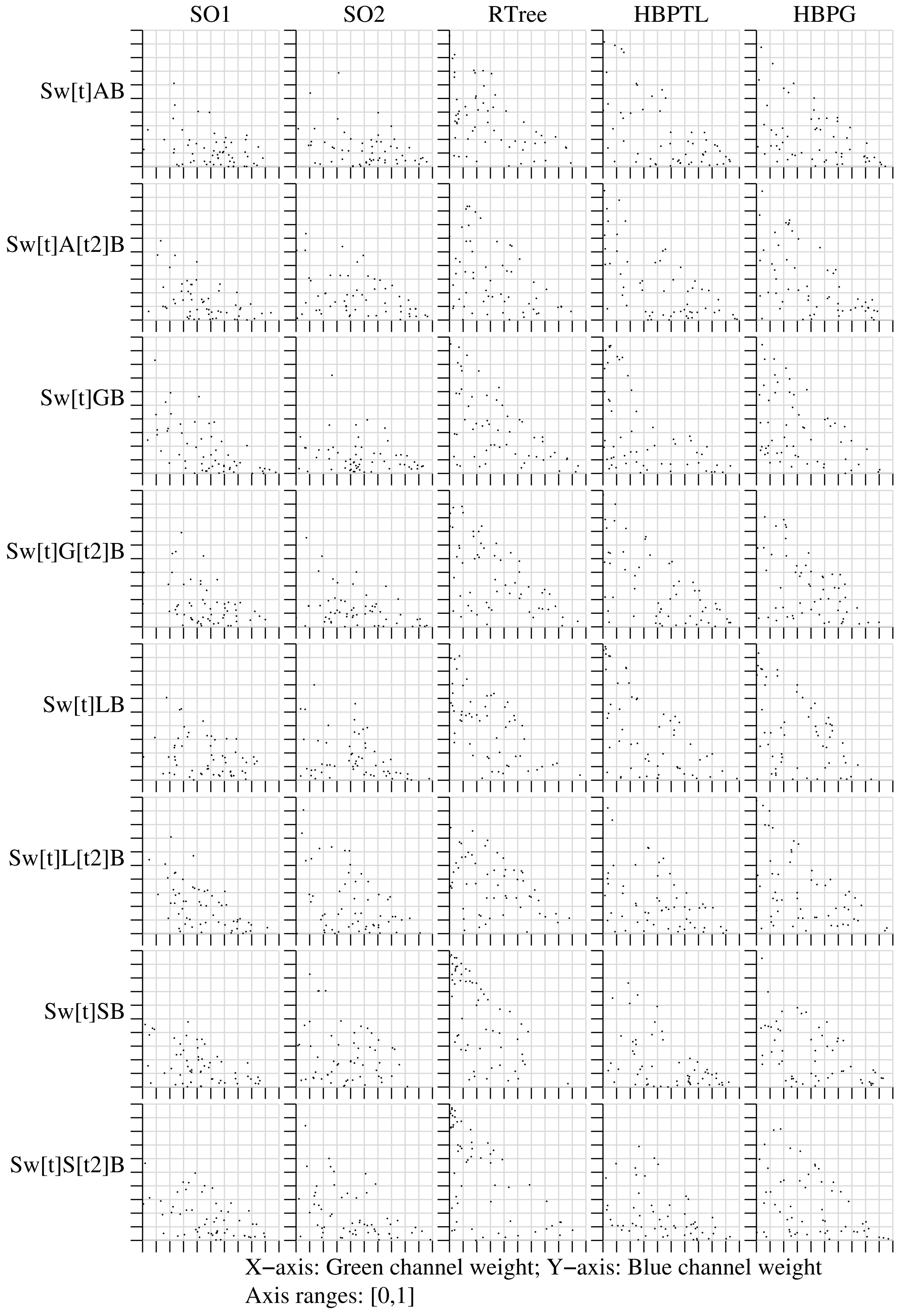
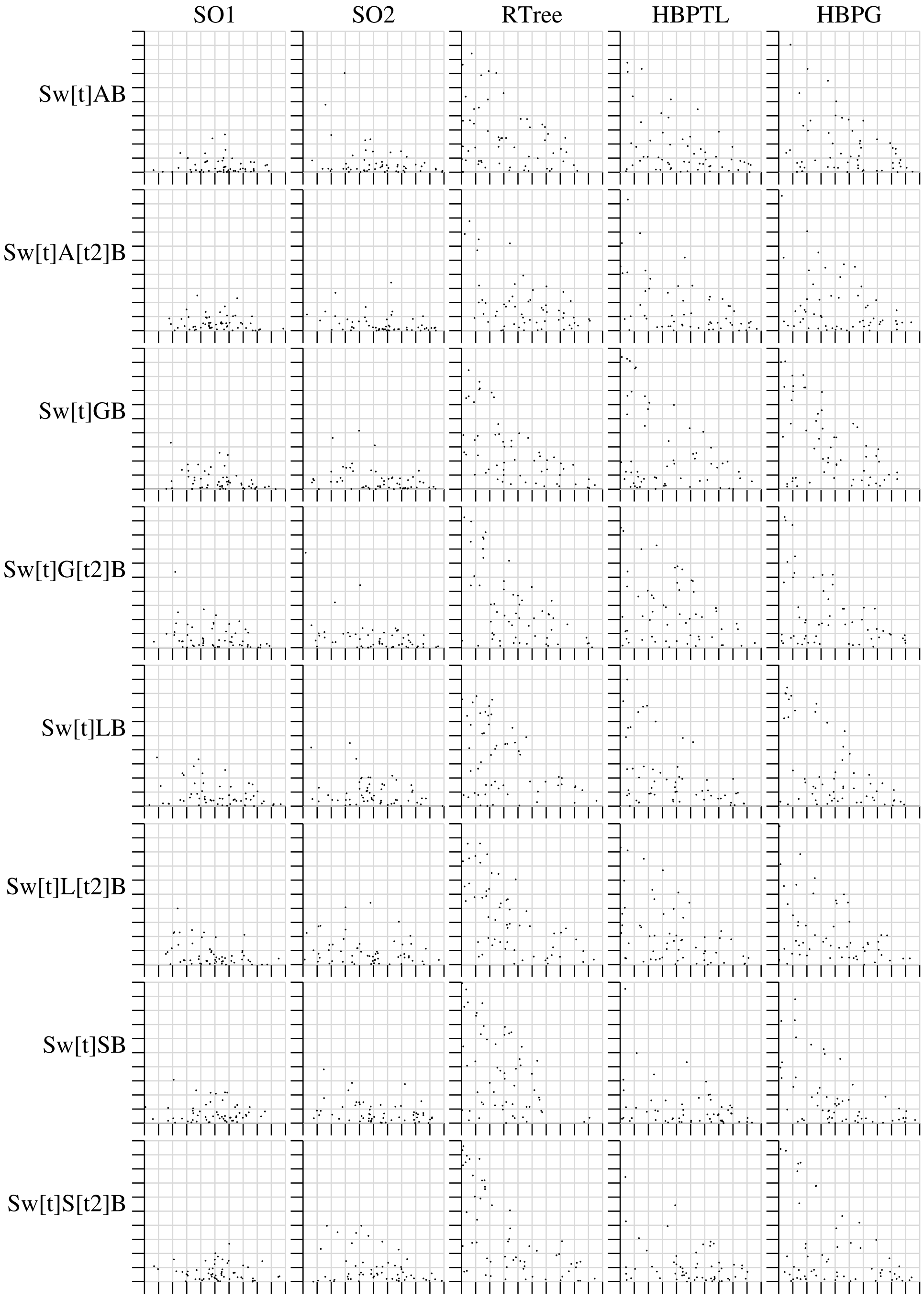


Figure F.11: Scatter plots of the blue and green weights of the weighted sum channel aggregate for all Sw[t]*B form match cost functions on the data set with low magnitude noise.



X−axis: Green channel weight; Y−axis: Blue channel weight
Axis ranges: [0,1]

Figure F.12: Scatter plots of the blue and green weights of the weighted sum channel aggregate for all Sw[t]*B form match cost functions on the data set with high magnitude noise.

Appendix G

Spatial Aggregate Rankings For Each Global Stereopsis Framework

Noise→	None				Low Magnitude				High Magnitude			
Spatial Aggregate	**S[t]AD**	**MxA[t]D**	**MxG[t]D**	**Sq[t]GD**	**S[t]AD**	**MxA[t]D**	**MxG[t]D**	**Sq[t]GD**	**S[t]AD**	**MxA[t]D**	**MxG[t]D**	**Sq[t]GD**
M ($r = 1$)	8	5	6	6	7	8	9	7	9	9	11	8
M ($r = 3$)	9	6	8	7	6	7	8	6	7	6	8	5
M ($r = 5$)	10	7	9	8	7	8	8	7	6	5	7	4
M ($r = 7$)	11	8	10	9	8	9	9	8	6	6	7	5
M ($r = 9$)	13	10	12	11	10	10	10	9	7	6	8	5
M ($r = 11$)	15	12	14	13	12	12	12	11	7	7	9	7
M ($r = 13$)	17	14	16	15	13	14	13	13	7	8	10	8
M ($r = 15$)	18	15	17	16	14	15	14	14	8	9	11	8
Ws ($r = 1$)	6	5	6	6	8	9	9	8	11	11	13	10
Ws ($r = 3$)	5	5	6	6	6	7	7	6	7	6	8	4
Ws ($r = 5$)	7	6	7	7	6	7	7	6	5	5	7	4
Ws ($r = 7$)	10	7	9	8	7	8	7	7	5	5	5	4
Ws ($r = 9$)	11	8	10	9	8	9	9	7	5	5	7	4
Ws ($r = 11$)	12	9	11	10	9	10	10	8	5	6	7	5
Ws ($r = 13$)	14	11	13	12	11	11	11	10	6	6	8	6
Ws ($r = 15$)	16	13	15	14	12	13	12	12	7	7	9	7
Wc ($r = 1$)	5	4	5	5	7	9	9	7	10	10	12	9
Wc ($r = 3$)	3	2	2	3	5	5	5	5	7	5	7	4
Wc ($r = 5$)	1	1	1	1	2	3	3	2	5	4	4	3
Wc ($r = 7$)	1	1	1	1	2	2	2	2	3	3	3	2

Table G.1: Over all rankings of each spatial aggregate for each base match cost function at 95% confidence for each data set in the SO1 framework, Table 1 of 2. Rankings are oriented vertically.

Noise→	**None**				**Low Magnitude**				**High Magnitude**			
Spatial Aggregate	**S[t]AD**	**MxA[t]D**	**MxG[t]D**	**Sq[t]GD**	**S[t]AD**	**MxA[t]D**	**MxG[t]D**	**Sq[t]GD**	**S[t]AD**	**MxA[t]D**	**MxG[t]D**	**Sq[t]GD**
Wc ($r = 9$)	1	1	1	1	1	1	2	1	2	2	2	2
Wc ($r = 11$)	2	2	2	2	1	1	2	1	2	2	1	1
Wc ($r = 13$)	2	2	2	2	2	1	3	2	2	2	1	2
Wc ($r = 15$)	4	3	3	3	2	2	3	2	2	2	1	1
B ($r = 1$)	6	5	6	6	8	10	10	8	12	12	14	11
B ($r = 3$)	4	3	2	3	5	6	6	5	7	6	8	4
B ($r = 5$)	2	2	1	2	3	3	4	3	5	4	4	3
B ($r = 7$)	1	1	1	1	2	2	3	2	3	3	3	2
B ($r = 9$)	1	1	1	1	1	1	2	1	2	2	1	2
B ($r = 11$)	1	1	1	1	1	1	1	1	1	2	1	1
B ($r = 13$)	1	1	1	1	1	1	1	1	1	1	1	1
B ($r = 15$)	2	2	1	2	1	1	1	1	1	1	1	1
Wa ($r = 1$)	10	6	8	7	12	11	11	10	13	13	15	12
Wa ($r = 3$)	5	4	5	5	6	7	7	6	9	8	11	8
Wa ($r = 5$)	4	3	4	4	4	4	5	4	7	5	7	4
Wa ($r = 7$)	4	3	3	3	2	3	4	2	5	4	6	3
Wa ($r = 9$)	4	3	3	3	2	2	3	2	4	3	3	2
Wa ($r = 11$)	4	3	2	3	1	2	2	1	2	2	2	1
Wa ($r = 13$)	4	3	3	3	1	2	2	1	2	2	2	1
Wa ($r = 15$)	4	3	3	3	1	1	2	1	2	2	1	1

Table G.2: Over all rankings of each spatial aggregate for each base match cost function at 95% confidence for each data set in the SO1 framework, Table 2 of 2. Rankings are oriented vertically.

Noise→	**None**				**Low Magnitude**				**High Magnitude**			
Spatial Aggregate	**S[t]AD**	**MxA[t]D**	**MxG[t]D**	**Sq[t]GD**	**S[t]AD**	**MxA[t]D**	**MxG[t]D**	**Sq[t]GD**	**S[t]AD**	**MxA[t]D**	**MxG[t]D**	**Sq[t]GD**
M ($r = 1$)	5	7	6	7	6	7	8	7	8	9	7	8
M ($r = 3$)	6	8	7	8	5	7	7	7	5	7	6	7
M ($r = 5$)	8	9	8	10	7	8	8	8	5	7	5	7
M ($r = 7$)	9	10	10	12	8	9	10	10	5	7	6	7
M ($r = 9$)	11	12	12	14	9	11	12	12	6	8	6	8
M ($r = 11$)	13	14	14	16	11	12	13	14	6	13	7	9
M ($r = 13$)	15	16	16	18	14	14	15	15	7	14	8	10
M ($r = 15$)	16	17	17	19	15	15	16	16	8	15	9	11
Ws ($r = 1$)	5	7	6	7	8	8	9	8	11	12	9	10
Ws ($r = 3$)	5	7	6	7	5	6	7	6	6	7	6	7
Ws ($r = 5$)	7	8	7	9	6	6	7	7	5	7	5	6
Ws ($r = 7$)	8	9	8	11	7	7	7	8	4	7	5	6
Ws ($r = 9$)	9	10	9	12	8	8	9	9	5	7	5	7
Ws ($r = 11$)	10	11	11	13	9	10	11	11	5	8	6	8
Ws ($r = 13$)	12	13	13	15	10	11	12	13	6	9	6	8
Ws ($r = 15$)	14	15	15	17	12	13	14	14	6	13	7	9
Wc ($r = 1$)	5	5	5	6	7	8	8	7	10	11	9	9
Wc ($r = 3$)	3	3	2	3	4	5	5	4	5	7	5	5
Wc ($r = 5$)	1	2	1	2	3	2	3	3	4	6	3	3
Wc ($r = 7$)	1	1	1	1	2	2	2	2	2	3	2	3

Table G.3: Over all rankings of each spatial aggregate for each base match cost function at 95% confidence for each data set in the SO2 framework, Table 1 of 2. Rankings are oriented vertically.

Noise→	None				Low Magnitude				High Magnitude			
Spatial Aggregate	**S[t]AD**	**MxA[t]D**	**MxG[t]D**	**Sq[t]GD**	**S[t]AD**	**MxA[t]D**	**MxG[t]D**	**Sq[t]GD**	**S[t]AD**	**MxA[t]D**	**MxG[t]D**	**Sq[t]GD**
Wc ($r = 9$)	1	2	2	2	1	1	1	1	1	3	1	2
Wc ($r = 11$)	2	2	2	2	1	1	1	2	2	2	1	1
Wc ($r = 13$)	2	3	3	2	2	1	1	1	1	2	1	1
Wc ($r = 15$)	3	3	3	3	2	2	2	2	1	2	1	1
B ($r = 1$)	5	7	6	6	9	8	9	8	12	16	10	11
B ($r = 3$)	3	3	3	3	4	5	5	5	6	7	6	6
B ($r = 5$)	2	2	2	2	3	3	3	3	4	6	4	4
B ($r = 7$)	1	1	1	1	2	2	2	2	3	5	3	3
B ($r = 9$)	1	1	1	1	1	1	1	1	2	3	2	2
B ($r = 11$)	1	1	1	1	1	1	1	1	1	2	1	2
B ($r = 13$)	1	2	1	2	1	1	1	1	1	2	1	1
B ($r = 15$)	1	2	1	2	1	1	1	1	1	1	1	1
Wa ($r = 1$)	7	8	7	9	13	11	12	11	13	17	11	12
Wa ($r = 3$)	4	5	5	5	5	6	6	6	9	10	8	8
Wa ($r = 5$)	4	4	4	4	4	4	4	4	6	7	5	6
Wa ($r = 7$)	3	3	3	3	3	2	2	2	4	6	4	3
Wa ($r = 9$)	3	4	3	3	2	2	2	2	3	4	3	2
Wa ($r = 11$)	3	4	3	3	2	2	2	2	2	3	2	1
Wa ($r = 13$)	3	6	3	3	2	1	1	1	2	2	1	1
Wa ($r = 15$)	4	6	3	4	2	2	2	1	1	2	1	1

Table G.4: Over all rankings of each spatial aggregate for each base match cost function at 95% confidence for each data set in the SO2 framework, Table 2 of 2. Rankings are oriented vertically.

Noise→	**None**				**Low Magnitude**				**High Magnitude**			
Spatial Aggregate	**S[t]AD**	**MxA[t]D**	**MxG[t]D**	**Sq[t]GD**	**S[t]AD**	**MxA[t]D**	**MxG[t]D**	**Sq[t]GD**	**S[t]AD**	**MxA[t]D**	**MxG[t]D**	**Sq[t]GD**
M ($r = 1$)	1	1	1	1	1	1	1	1	1	1	1	1
M ($r = 3$)	1	1	1	1	1	1	1	1	1	1	1	1
M ($r = 5$)	1	1	1	1	1	1	1	1	1	1	1	1
M ($r = 7$)	1	1	1	1	1	1	1	1	1	1	1	1
M ($r = 9$)	1	1	1	1	1	1	1	1	1	1	1	1
M ($r = 11$)	1	1	1	1	1	1	1	1	1	1	1	1
M ($r = 13$)	1	1	1	1	1	1	1	1	1	1	1	1
M ($r = 15$)	1	1	1	1	1	1	1	1	1	1	1	1
Ws ($r = 1$)	1	1	1	1	1	1	1	1	1	1	1	1
Ws ($r = 3$)	1	1	1	1	1	1	1	1	1	1	1	1
Ws ($r = 5$)	1	1	1	1	1	1	1	1	1	1	1	1
Ws ($r = 7$)	1	1	1	1	1	1	1	1	1	1	1	1
Ws ($r = 9$)	1	1	1	1	1	1	1	1	1	1	1	1
Ws ($r = 11$)	1	1	1	1	1	1	1	1	1	1	1	1
Ws ($r = 13$)	1	1	1	1	1	1	1	1	1	1	1	1
Ws ($r = 15$)	1	1	1	1	1	1	1	1	1	1	1	1
Wc ($r = 1$)	1	1	1	1	1	1	1	1	1	1	1	1
Wc ($r = 3$)	1	1	1	1	1	1	1	1	1	1	1	1
Wc ($r = 5$)	1	1	1	1	1	1	1	1	1	1	1	1
Wc ($r = 7$)	1	1	1	1	1	1	1	1	1	1	1	1

Table G.5: Over all rankings of each spatial aggregate for each base match cost function at 95% confidence for each data set in the RTree framework, Table 1 of 2. Rankings are oriented vertically.

Noise→	**None**				**Low Magnitude**				**High Magnitude**			
Spatial Aggregate	**S[t]AD**	**MxA[t]D**	**MxG[t]D**	**Sq[t]GD**	**S[t]AD**	**MxA[t]D**	**MxG[t]D**	**Sq[t]GD**	**S[t]AD**	**MxA[t]D**	**MxG[t]D**	**Sq[t]GD**
Wc ($r = 9$)	1	1	1	1	1	1	1	1	1	1	1	1
Wc ($r = 11$)	1	1	1	1	1	1	1	1	1	1	1	1
Wc ($r = 13$)	1	1	1	1	1	1	1	1	1	1	1	1
Wc ($r = 15$)	1	1	1	1	1	1	1	1	1	1	1	1
B ($r = 1$)	1	1	1	1	1	1	1	1	1	1	1	1
B ($r = 3$)	1	1	1	1	1	1	1	1	1	1	1	1
B ($r = 5$)	1	1	1	1	1	1	1	1	1	1	1	1
B ($r = 7$)	1	1	1	1	1	1	1	1	1	1	1	1
B ($r = 9$)	1	1	1	1	1	1	1	1	1	1	1	1
B ($r = 11$)	1	1	1	1	1	1	1	1	1	1	1	1
B ($r = 13$)	1	1	1	1	1	1	1	1	1	1	1	1
B ($r = 15$)	1	1	1	1	1	1	1	1	1	1	1	1
Wa ($r = 1$)	1	1	1	1	1	1	1	1	1	1	1	1
Wa ($r = 3$)	1	1	1	1	1	1	1	1	1	1	1	1
Wa ($r = 5$)	1	1	1	1	1	1	1	1	1	1	1	1
Wa ($r = 7$)	1	1	1	1	1	1	1	1	1	1	1	1
Wa ($r = 9$)	1	1	1	1	1	1	1	1	1	1	1	1
Wa ($r = 11$)	1	1	1	1	1	1	1	1	1	1	1	1
Wa ($r = 13$)	1	1	1	1	1	1	1	1	1	1	1	1
Wa ($r = 15$)	1	1	1	1	1	1	1	1	1	1	1	1

Table G.6: Over all rankings of each spatial aggregate for each base match cost function at 95% confidence for each data set in the RTree framework, Table 2 of 2. Rankings are oriented vertically.

Noise→	**None**				**Low Magnitude**				**High Magnitude**			
Spatial Aggregate	**S[t]AD**	**MxA[t]D**	**MxG[t]D**	**Sq[t]GD**	**S[t]AD**	**MxA[t]D**	**MxG[t]D**	**Sq[t]GD**	**S[t]AD**	**MxA[t]D**	**MxG[t]D**	**Sq[t]GD**
M ($r = 1$)	5	7	9	6	5	5	6	6	6	5	6	5
M ($r = 3$)	6	8	10	9	6	5	6	7	5	5	5	4
M ($r = 5$)	9	10	11	10	8	6	7	8	4	6	6	5
M ($r = 7$)	10	12	12	12	9	8	8	9	5	6	6	6
M ($r = 9$)	12	14	14	14	11	10	10	11	6	8	7	6
M ($r = 11$)	14	16	16	16	12	11	12	13	6	9	8	7
M ($r = 13$)	15	17	17	17	13	12	13	15	8	10	9	7
M ($r = 15$)	16	18	18	18	14	13	14	16	9	11	10	8
Ws ($r = 1$)	5	6	7	6	5	5	6	6	6	6	6	5
Ws ($r = 3$)	7	9	7	8	5	5	5	6	4	5	5	4
Ws ($r = 5$)	6	8	8	7	5	5	6	6	4	5	6	5
Ws ($r = 7$)	8	10	11	10	7	6	6	8	5	6	6	5
Ws ($r = 9$)	10	11	12	11	8	7	7	9	5	6	6	5
Ws ($r = 11$)	11	13	13	13	10	9	9	10	5	7	7	6
Ws ($r = 13$)	13	15	15	15	11	10	11	12	6	7	7	6
Ws ($r = 15$)	14	16	16	16	12	11	12	14	7	9	8	7
Wc ($r = 1$)	4	5	5	5	5	5	6	6	6	6	6	5
Wc ($r = 3$)	2	3	3	2	3	3	4	4	3	4	4	4
Wc ($r = 5$)	2	2	2	2	2	2	2	2	2	3	3	3
Wc ($r = 7$)	1	2	1	1	1	1	1	1	1	2	2	2

Table G.7: Over all rankings of each spatial aggregate for each base match cost function at 95% confidence for each data set in the HBPTL framework, Table 1 of 2. Rankings are oriented vertically.

Noise→	**None**				**Low Magnitude**				**High Magnitude**			
Spatial Aggregate	**S[t]AD**	**MxA[t]D**	**MxG[t]D**	**Sq[t]GD**	**S[t]AD**	**MxA[t]D**	**MxG[t]D**	**Sq[t]GD**	**S[t]AD**	**MxA[t]D**	**MxG[t]D**	**Sq[t]GD**
Wc ($r = 9$)	1	2	2	1	1	1	1	1	1	2	2	1
Wc ($r = 11$)	2	2	2	2	1	1	1	1	1	1	2	1
Wc ($r = 13$)	2	3	2	2	1	1	1	2	1	1	1	1
Wc ($r = 15$)	3	3	3	3	2	2	2	2	1	1	2	1
B ($r = 1$)	5	6	6	5	5	5	6	6	6	6	8	6
B ($r = 3$)	3	3	3	3	3	3	4	4	4	4	5	4
B ($r = 5$)	2	2	2	2	2	2	3	3	2	3	3	2
B ($r = 7$)	1	2	1	1	1	1	2	2	1	2	3	2
B ($r = 9$)	1	1	1	1	1	1	1	1	1	2	2	2
B ($r = 11$)	1	1	1	1	1	1	1	1	1	1	1	1
B ($r = 13$)	1	2	1	1	1	1	1	1	1	1	1	1
B ($r = 15$)	2	2	2	2	1	1	1	1	1	1	1	1
Wa ($r = 1$)	5	6	7	6	8	6	7	8	8	8	9	7
Wa ($r = 3$)	4	5	5	4	4	4	5	5	5	5	6	5
Wa ($r = 5$)	3	4	4	3	3	3	4	4	3	5	5	4
Wa ($r = 7$)	3	4	3	3	2	2	3	2	3	2	3	2
Wa ($r = 9$)	3	3	3	3	2	2	2	2	2	2	2	1
Wa ($r = 11$)	3	3	3	3	1	2	2	2	1	2	2	1
Wa ($r = 13$)	3	4	3	3	2	2	2	1	1	1	2	1
Wa ($r = 15$)	3	4	3	3	2	2	2	2	1	1	2	1

Table G.8: Over all rankings of each spatial aggregate for each base match cost function at 95% confidence for each data set in the HBPTL framework, Table 2 of 2. Rankings are oriented vertically.

Noise→	**None**				**Low Magnitude**				**High Magnitude**			
Spatial Aggregate	**S[t]AD**	**MxA[t]D**	**MxG[t]D**	**Sq[t]GD**	**S[t]AD**	**MxA[t]D**	**MxG[t]D**	**Sq[t]GD**	**S[t]AD**	**MxA[t]D**	**MxG[t]D**	**Sq[t]GD**
M ($r = 1$)	5	6	5	5	5	5	5	5	5	5	5	4
M ($r = 3$)	6	7	6	6	5	5	5	5	4	4	4	3
M ($r = 5$)	7	9	8	8	6	6	6	6	4	4	4	4
M ($r = 7$)	8	10	9	9	7	7	8	7	4	5	4	4
M ($r = 9$)	10	12	11	11	8	8	9	8	6	5	5	5
M ($r = 11$)	12	14	12	13	10	10	11	10	7	6	6	6
M ($r = 13$)	13	15	14	14	12	12	13	12	8	7	8	7
M ($r = 15$)	14	16	15	15	13	13	14	13	9	8	9	8
Ws ($r = 1$)	5	5	4	4	5	6	6	5	6	5	5	4
Ws ($r = 3$)	5	6	5	5	5	5	5	5	4	4	4	3
Ws ($r = 5$)	6	8	7	7	5	5	5	5	4	4	4	3
Ws ($r = 7$)	7	9	8	8	5	6	6	6	4	4	4	3
Ws ($r = 9$)	8	10	9	9	7	6	7	7	5	4	5	4
Ws ($r = 11$)	9	11	10	10	7	8	8	7	5	5	5	4
Ws ($r = 13$)	11	13	12	12	9	9	10	9	6	6	6	5
Ws ($r = 15$)	12	14	13	13	11	11	12	11	7	6	7	6
Wc ($r = 1$)	4	5	4	4	5	5	5	6	6	5	5	4
Wc ($r = 3$)	2	4	2	3	4	3	3	4	4	4	3	3
Wc ($r = 5$)	1	2	2	2	3	2	2	2	3	3	3	2
Wc ($r = 7$)	1	2	2	1	2	1	1	1	2	2	2	1

Table G.9: Over all rankings of each spatial aggregate for each base match cost function at 95% confidence for each data set in the HBPG framework, Table 1 of 2. Rankings are oriented vertically.

Noise→	None				Low Magnitude				High Magnitude			
Spatial Aggregate	**S[t]AD**	**MxA[t]D**	**MxG[t]D**	**Sq[t]GD**	**S[t]AD**	**MxA[t]D**	**MxG[t]D**	**Sq[t]GD**	**S[t]AD**	**MxA[t]D**	**MxG[t]D**	**Sq[t]GD**
Wc ($r = 9$)	1	2	1	1	1	1	1	1	2	2	1	1
Wc ($r = 11$)	2	2	2	2	2	1	1	1	1	1	1	1
Wc ($r = 13$)	2	2	2	2	2	1	1	1	1	1	1	1
Wc ($r = 15$)	2	4	3	2	2	2	2	2	2	2	2	2
B ($r = 1$)	5	5	5	5	6	6	6	6	7	6	6	5
B ($r = 3$)	2	4	3	3	4	3	4	4	5	4	4	3
B ($r = 5$)	2	3	2	2	3	2	2	3	3	3	3	2
B ($r = 7$)	1	2	1	1	2	2	1	2	2	2	2	2
B ($r = 9$)	1	1	1	1	2	1	1	1	2	2	2	1
B ($r = 11$)	1	1	1	1	1	1	1	1	1	1	1	1
B ($r = 13$)	1	2	1	1	1	1	1	1	1	1	1	1
B ($r = 15$)	1	2	2	1	1	1	1	1	1	1	1	1
Wa ($r = 1$)	6	6	6	6	7	7	8	7	7	6	8	6
Wa ($r = 3$)	4	5	4	4	5	5	5	5	6	5	5	4
Wa ($r = 5$)	3	4	3	3	4	4	4	4	4	4	4	3
Wa ($r = 7$)	3	4	3	3	3	3	2	3	4	3	3	2
Wa ($r = 9$)	3	4	3	2	2	2	2	2	4	3	2	2
Wa ($r = 11$)	3	4	3	3	2	2	2	2	2	2	2	1
Wa ($r = 13$)	3	4	3	3	2	2	2	2	2	2	2	1
Wa ($r = 15$)	3	4	3	3	3	2	2	2	2	2	1	1

Table G.10: Over all rankings of each spatial aggregate for each base match cost function at 95% confidence for each data set in the HBPG framework, Table 2 of 2. Rankings are oriented vertically.

Appendix H

Graphs Of Mean Accuracy Improvement For Match Cost Functions As The Spatial Aggregate Radius Increases.

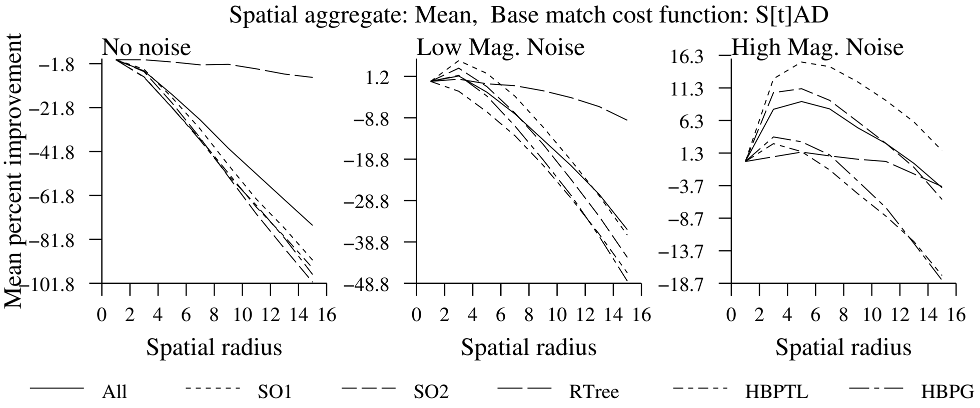


Figure H.1: Mean accuracy improvement of the MS[t]AD match cost function as the spatial radius, r, is increased.

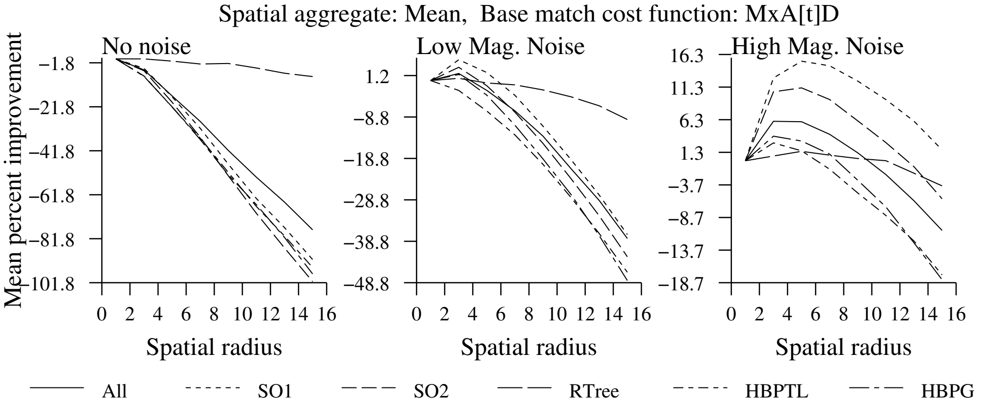


Figure H.2: Mean accuracy improvement of the MMxA[t]D match cost function as the spatial radius, r, is increased.

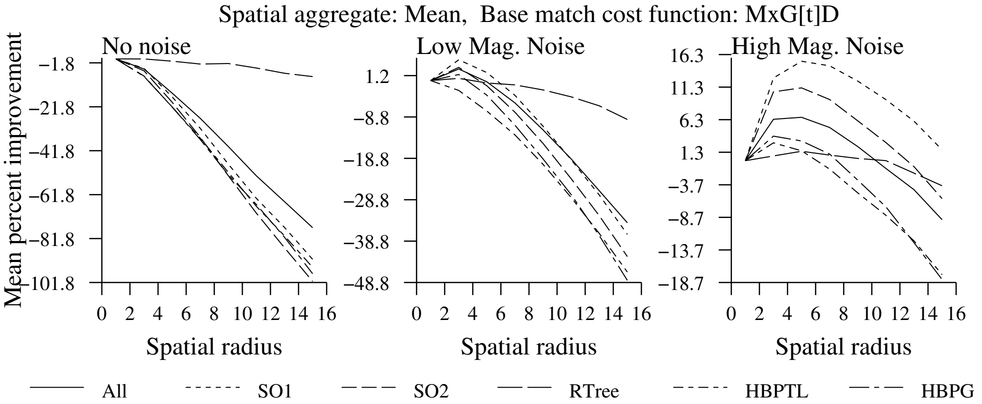


Figure H.3: Mean accuracy improvement of the MMxG[t]D match cost function as the spatial radius, r, is increased.

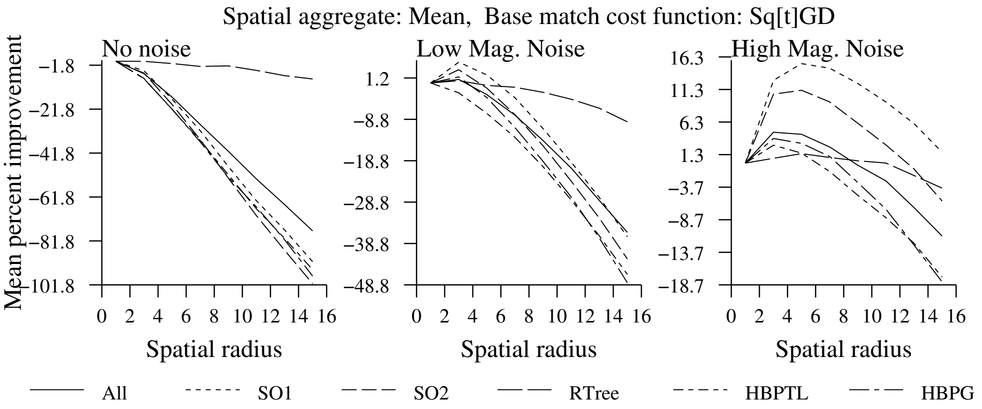


Figure H.4: Mean accuracy improvement of the MSq[t]GD match cost function as the spatial radius, r, is increased.

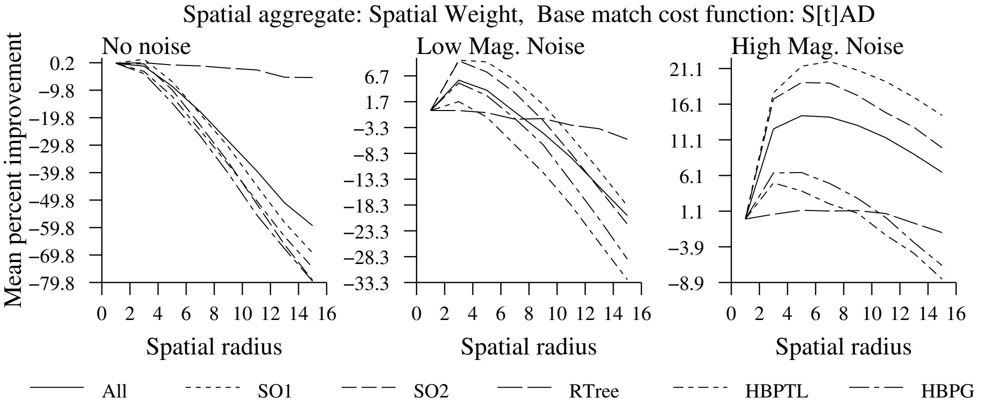


Figure H.5: Mean accuracy improvement of the WsS[t]AD match cost function as the spatial radius, r, is increased.

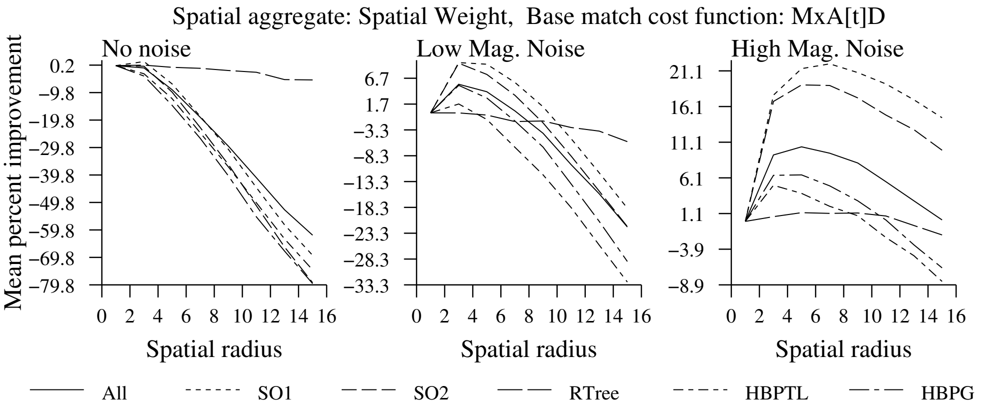


Figure H.6: Mean accuracy improvement of the WsMxA[t]D match cost function as the spatial radius, r, is increased.

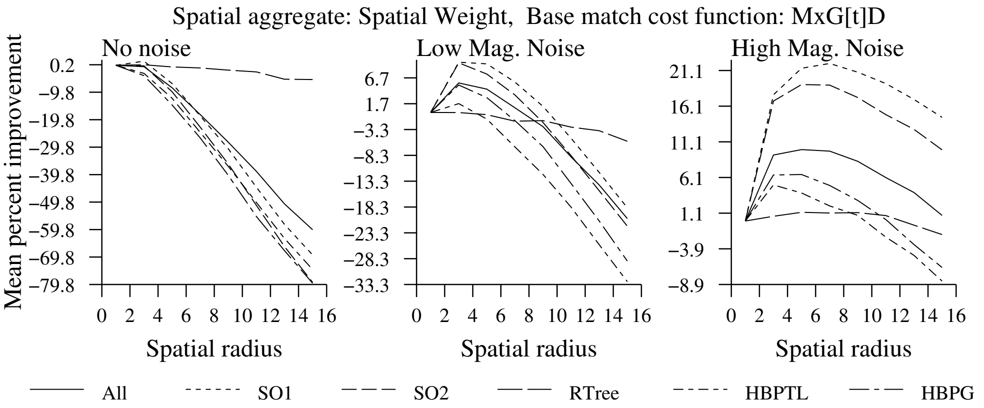


Figure H.7: Mean accuracy improvement of the WsMxG[t]D match cost function as the spatial radius, r, is increased.

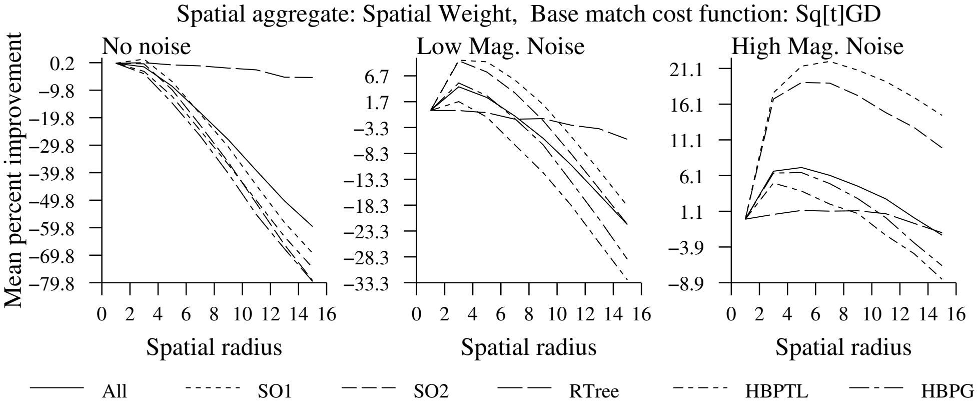


Figure H.8: Mean accuracy improvement of the WsSq[t]GD match cost function as the spatial radius, r, is increased.

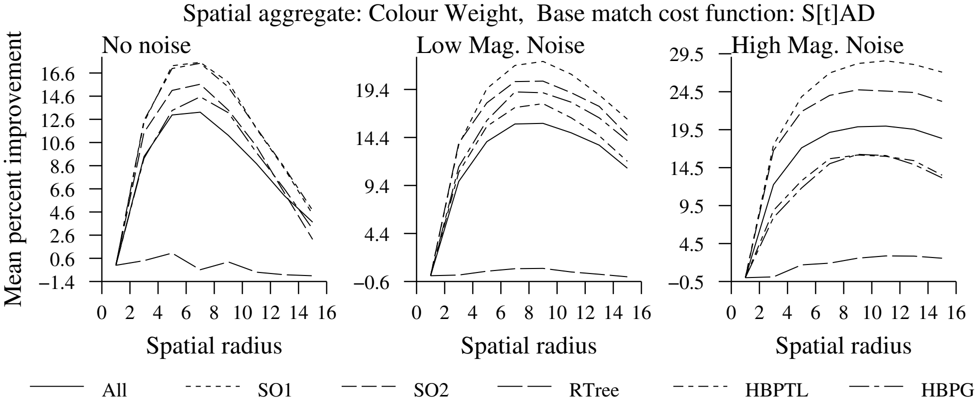


Figure H.9: Mean accuracy improvement of the WcS[t]AD match cost function as the spatial radius, r, is increased.

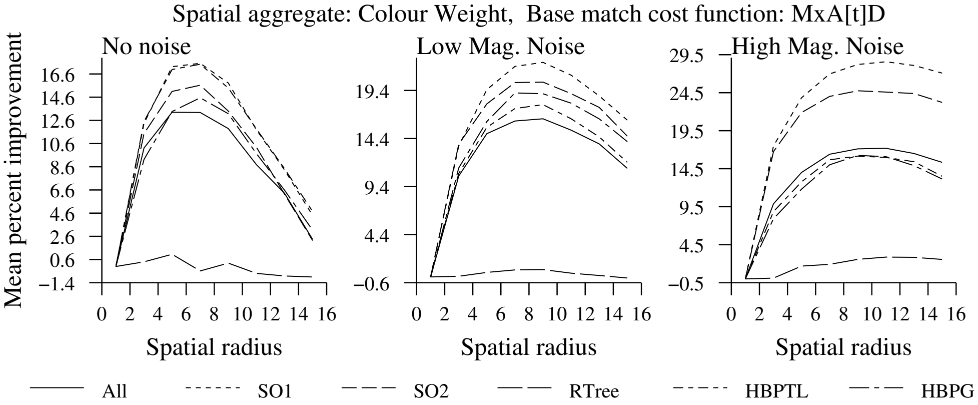


Figure H.10: Mean accuracy improvement of the WcMxA[t]D match cost function as the spatial radius, r, is increased.

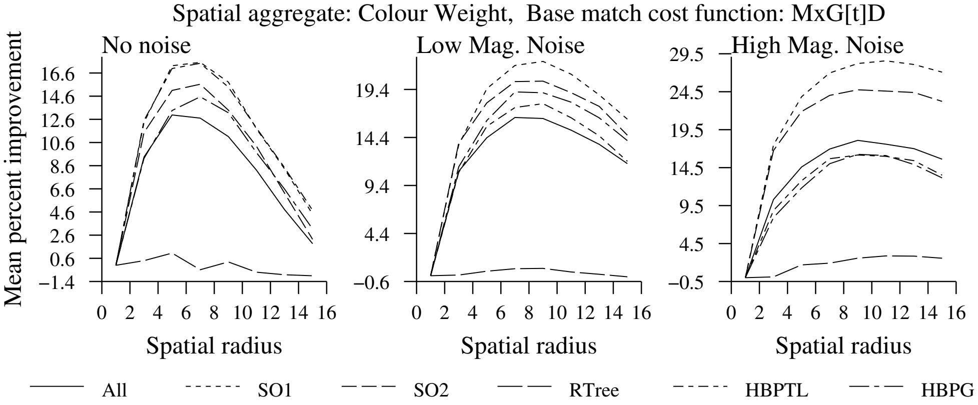


Figure H.11: Mean accuracy improvement of the WcMxG[t]D match cost function as the spatial radius, r, is increased.

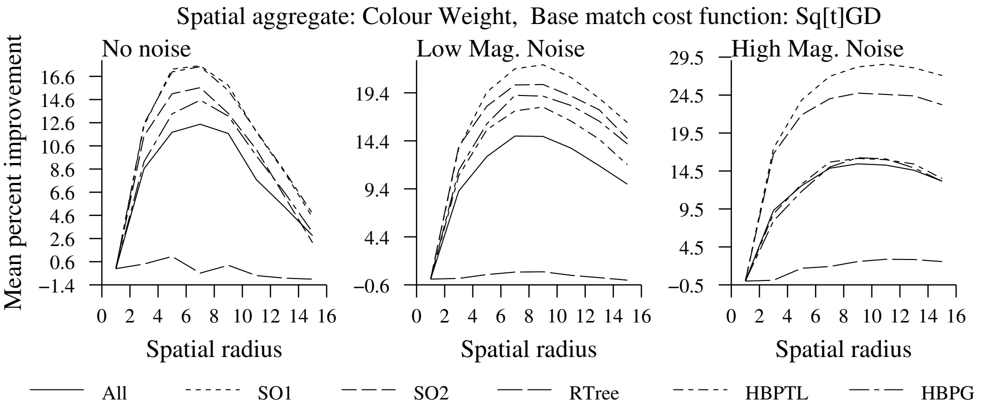


Figure H.12: Mean accuracy improvement of the WcSq[t]GD match cost function as the spatial radius, r, is increased.

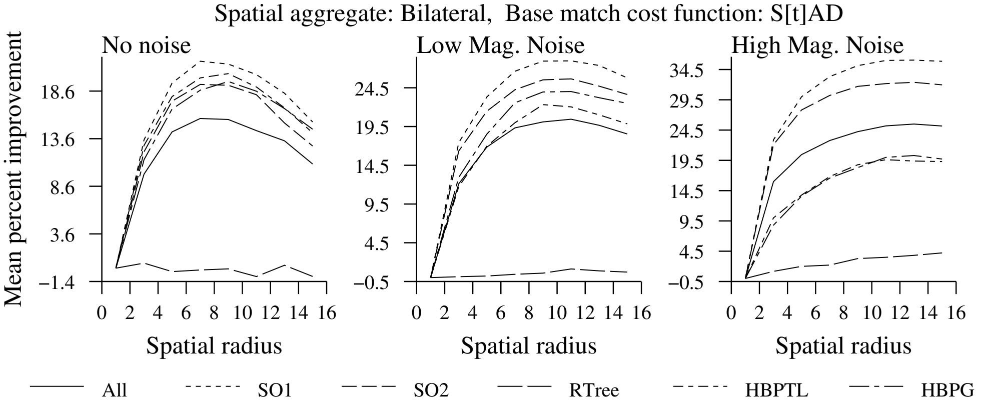


Figure H.13: Mean accuracy improvement of the BS[t]AD match cost function as the spatial radius, r, is increased.

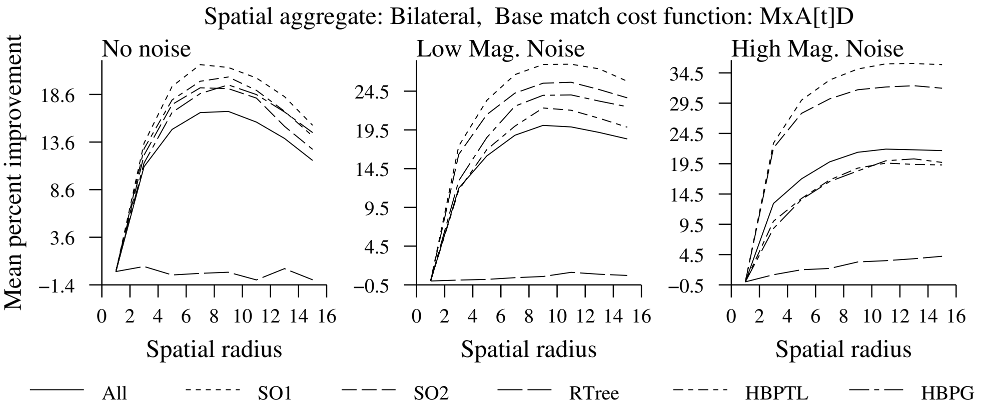


Figure H.14: Mean accuracy improvement of the BMxA[t]D match cost function as the spatial radius, r, is increased.

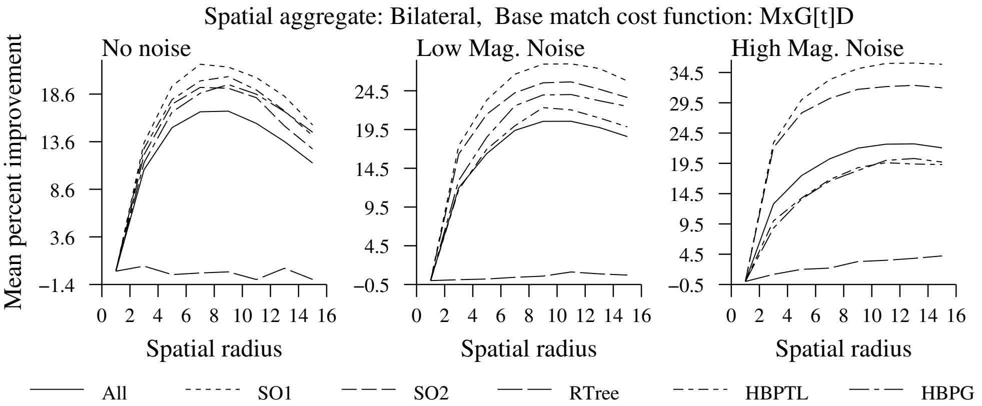


Figure H.15: Mean accuracy improvement of the BMxG[t]D match cost function as the spatial radius, r, is increased.

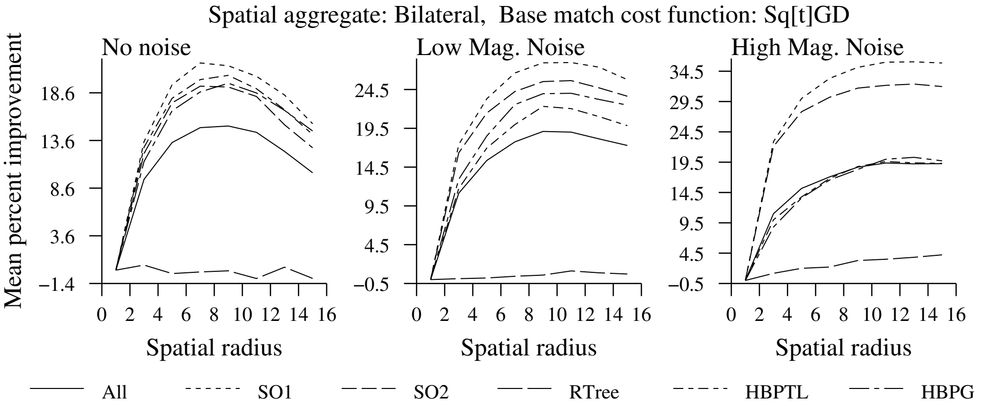


Figure H.16: Mean accuracy improvement of the BSq[t]GD match cost function as the spatial radius, r, is increased.

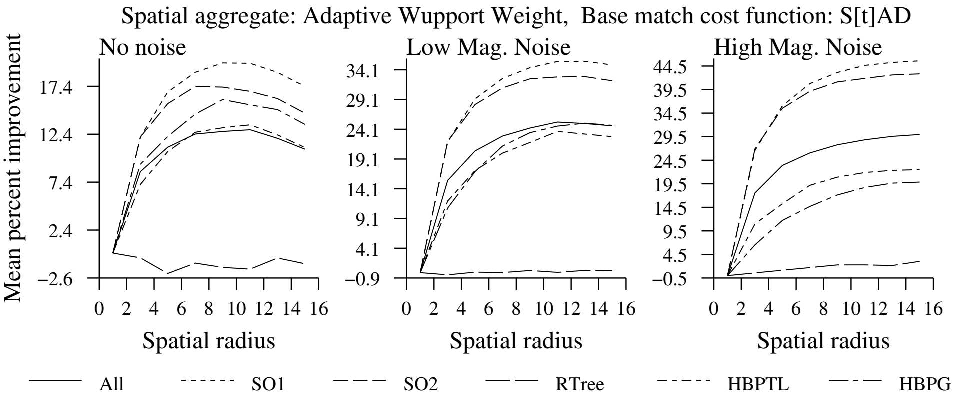


Figure H.17: Mean accuracy improvement of the WaS[t]AD match cost function as the spatial radius, r, is increased.

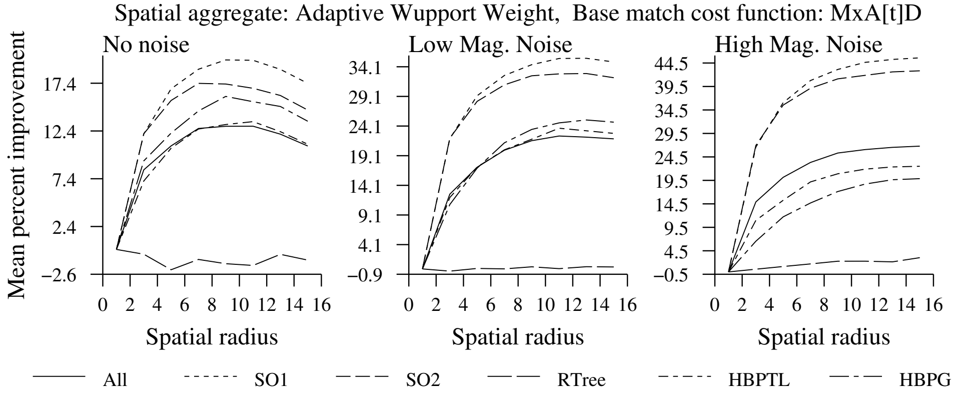


Figure H.18: Mean accuracy improvement of the WaMxA[t]D match cost function as the spatial radius, r, is increased.

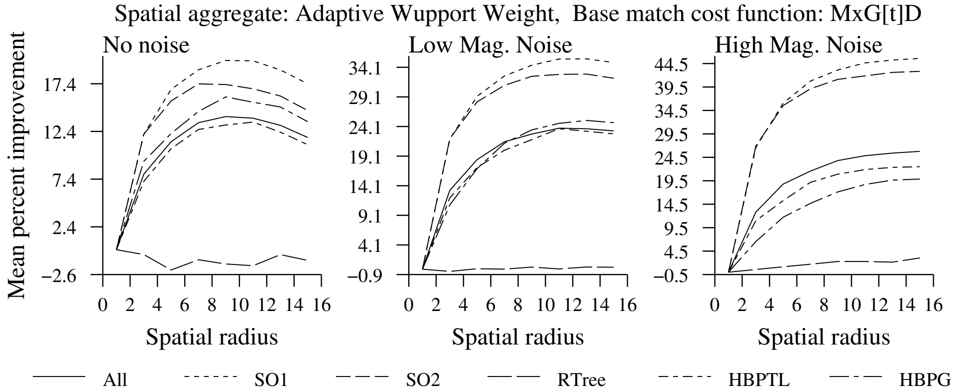


Figure H.19: Mean accuracy improvement of the WaMxG[t]D match cost function as the spatial radius, r, is increased.

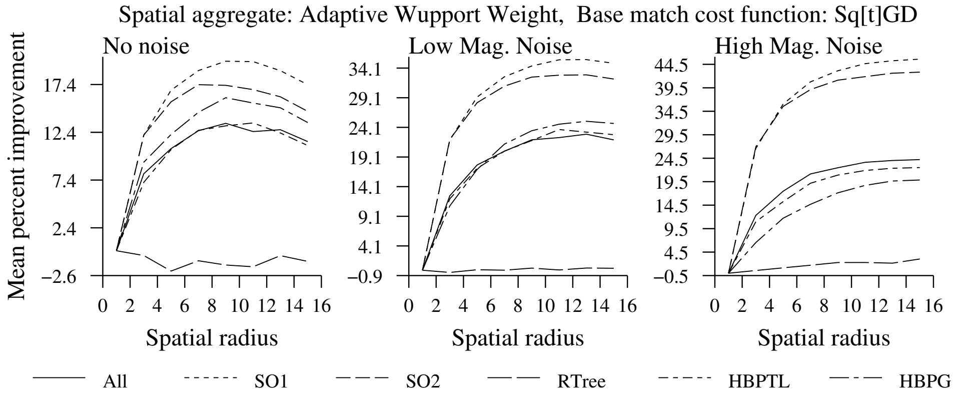


Figure H.20: Mean accuracy improvement of the WaSq[t]GD match cost function as the spatial radius, r, is increased.

Appendix I

Aggregate Improvement for Colour Spaces by Framework

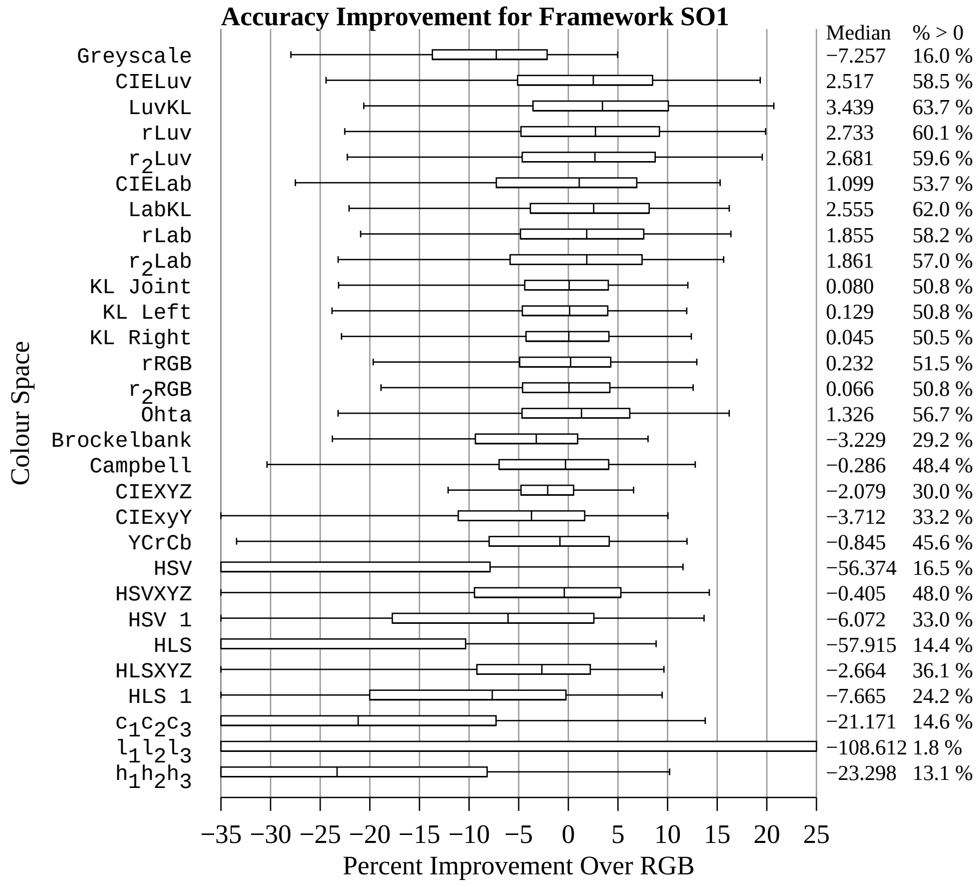


Figure I.1: Box plots of the over all aggregate percent accuracy improvement distributions for the framework for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

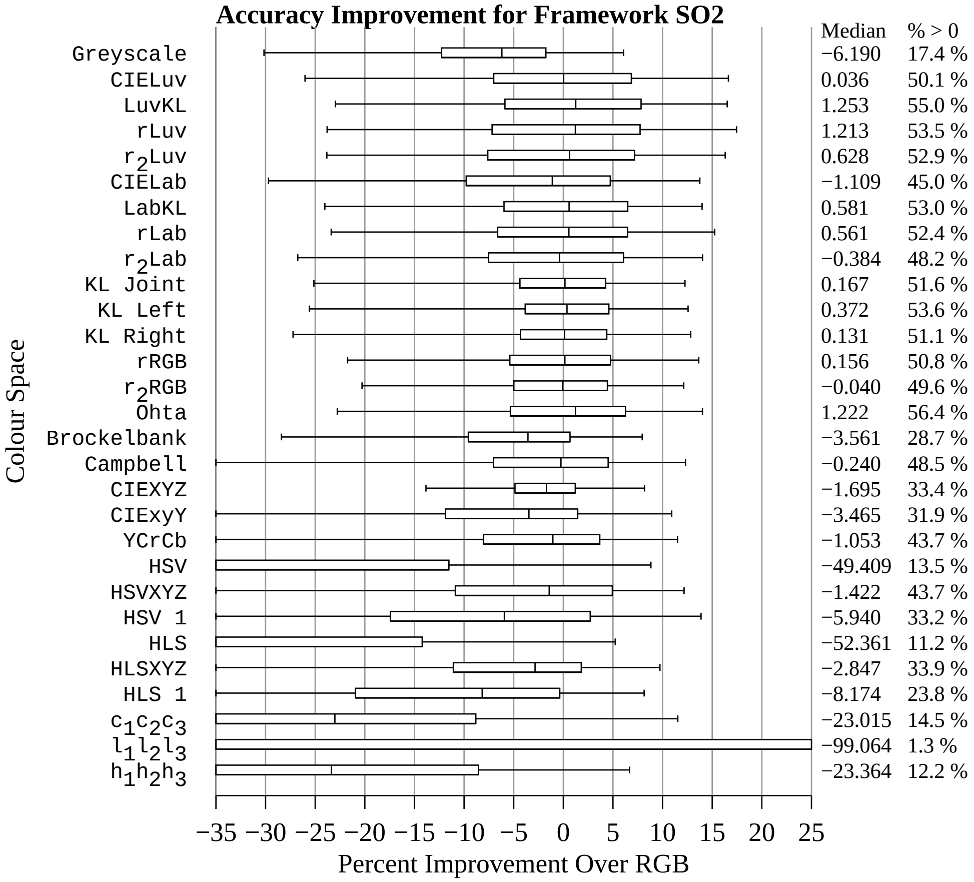


Figure I.2: Box plots of the over all aggregate percent accuracy improvement distributions for the framework for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

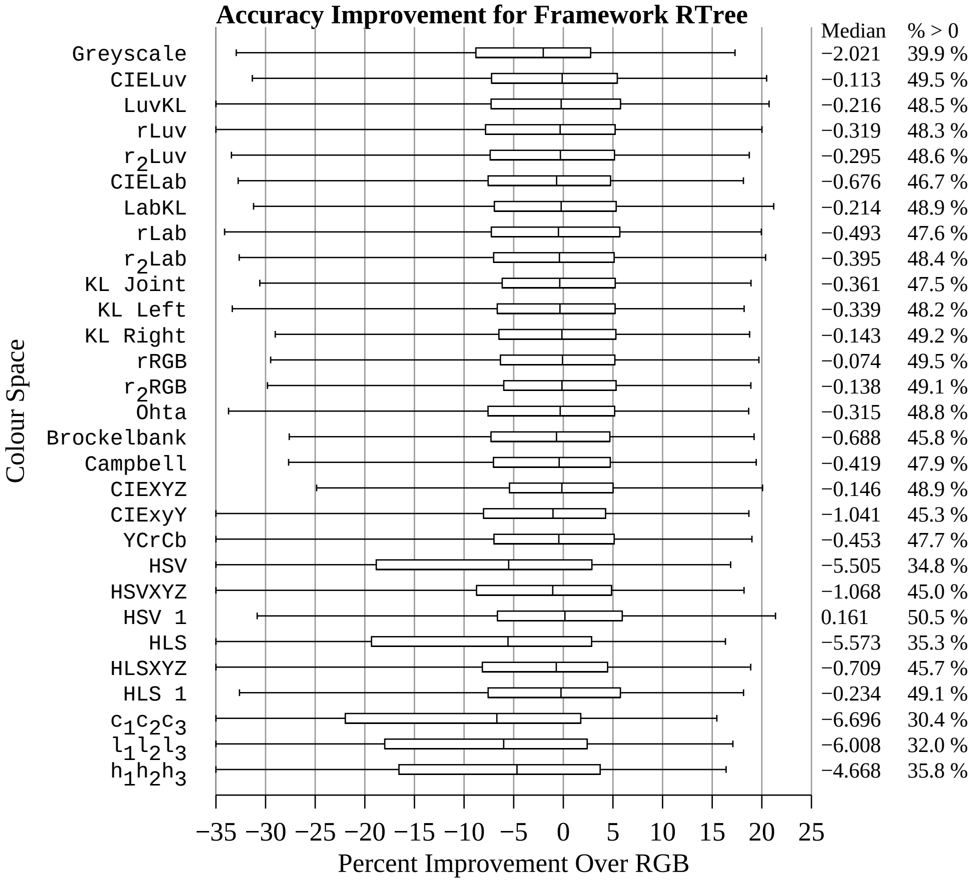


Figure I.3: Box plots of the over all aggregate percent accuracy improvement distributions for the framework for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

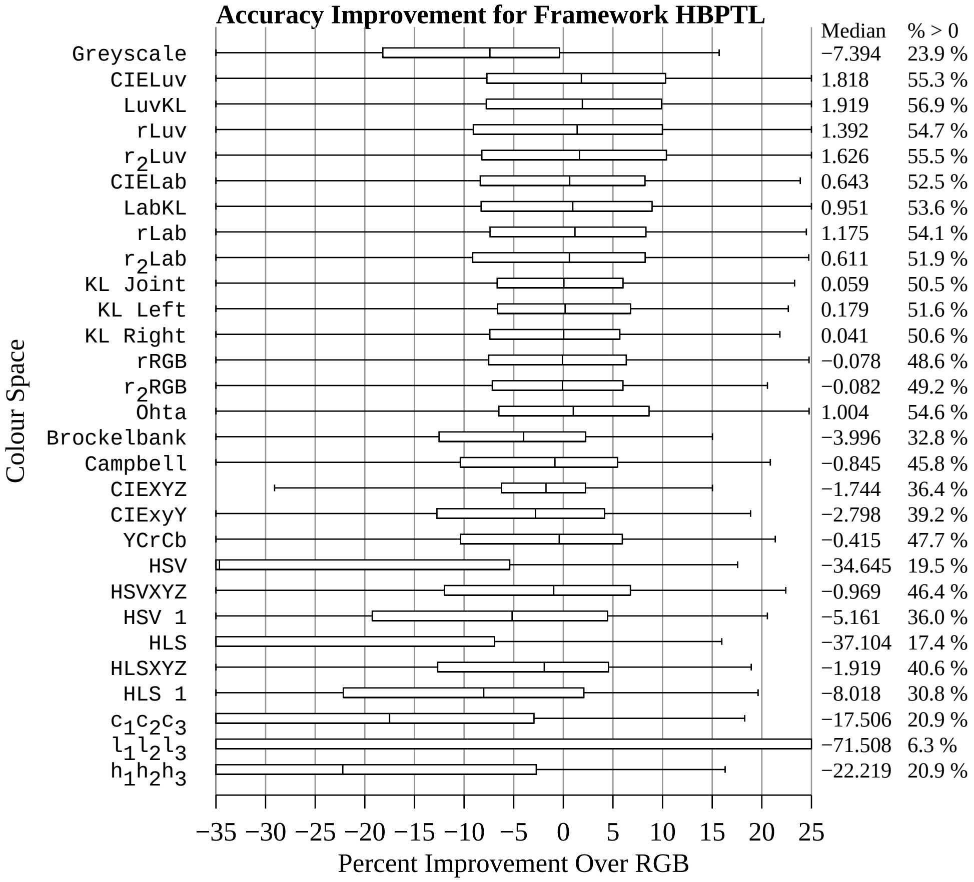


Figure I.4: Box plots of the over all aggregate percent accuracy improvement distributions for the framework for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

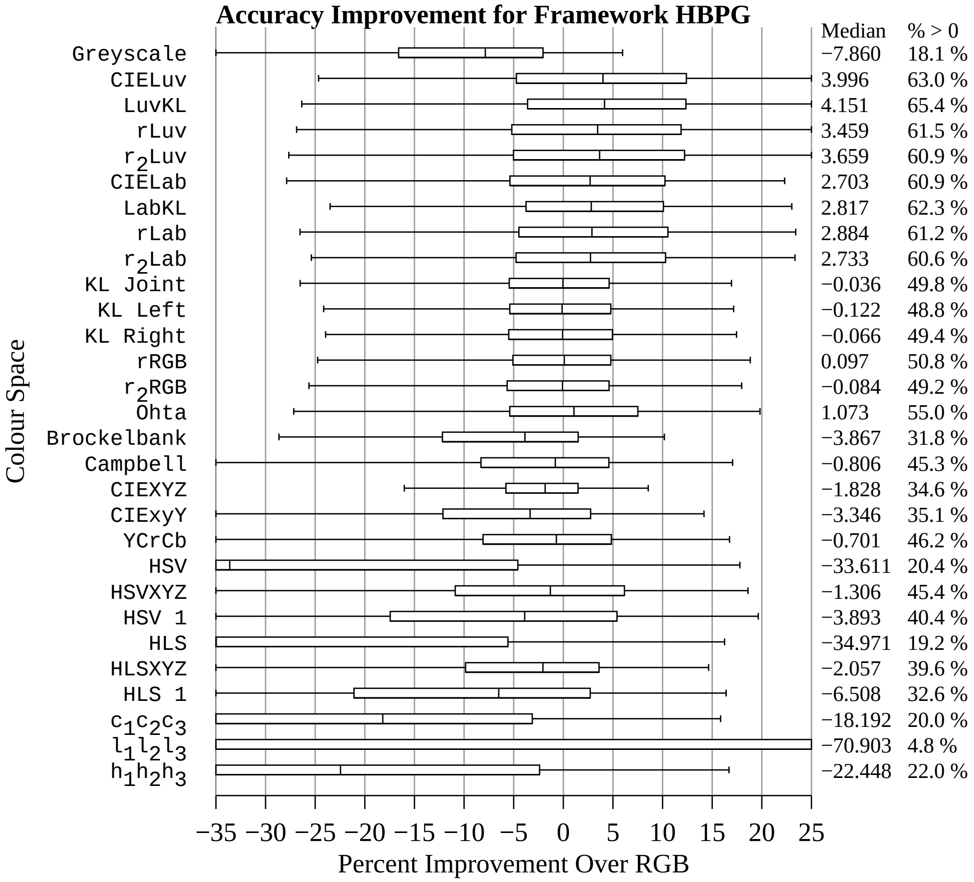


Figure I.5: Box plots of the over all aggregate percent accuracy improvement distributions for the framework for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

Appendix J

Aggregate Improvement for Colour Spaces by Match Cost Function

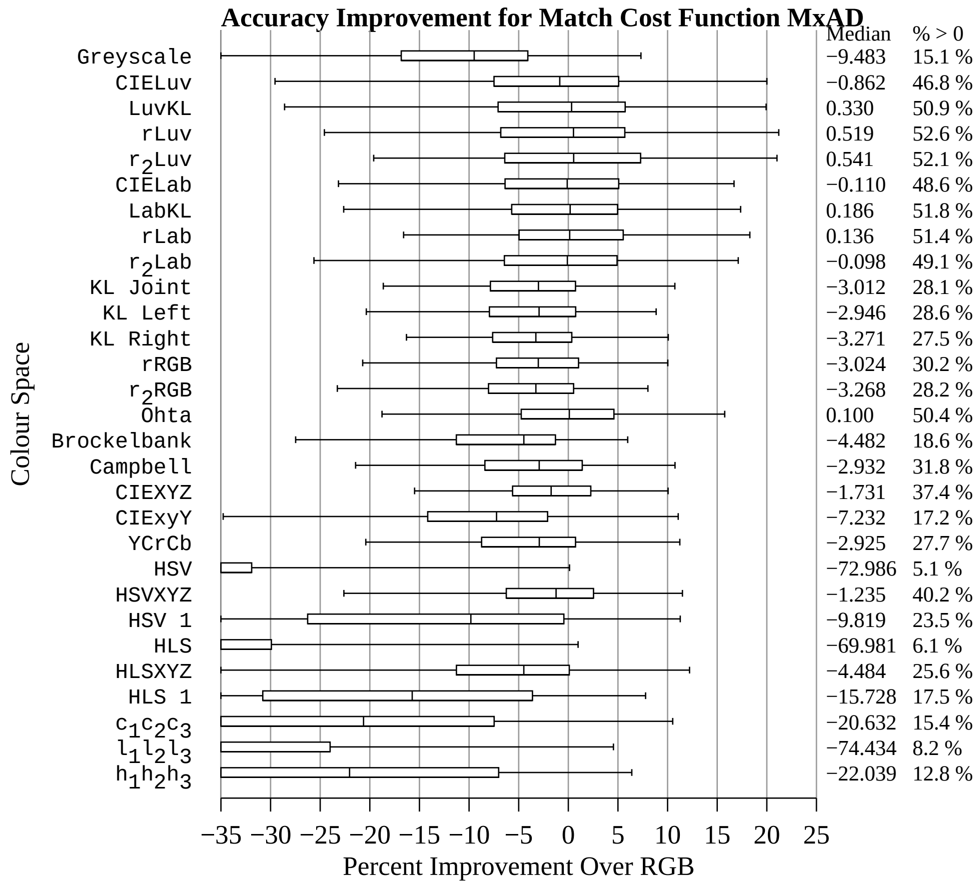


Figure J.1: Box plots of the over all aggregate percent accuracy improvement distributions for the MxAD match cost function for each colour representation. The plots indicate the fifth, 25th, 50th, 75th, and 95th percentiles of the distribution.

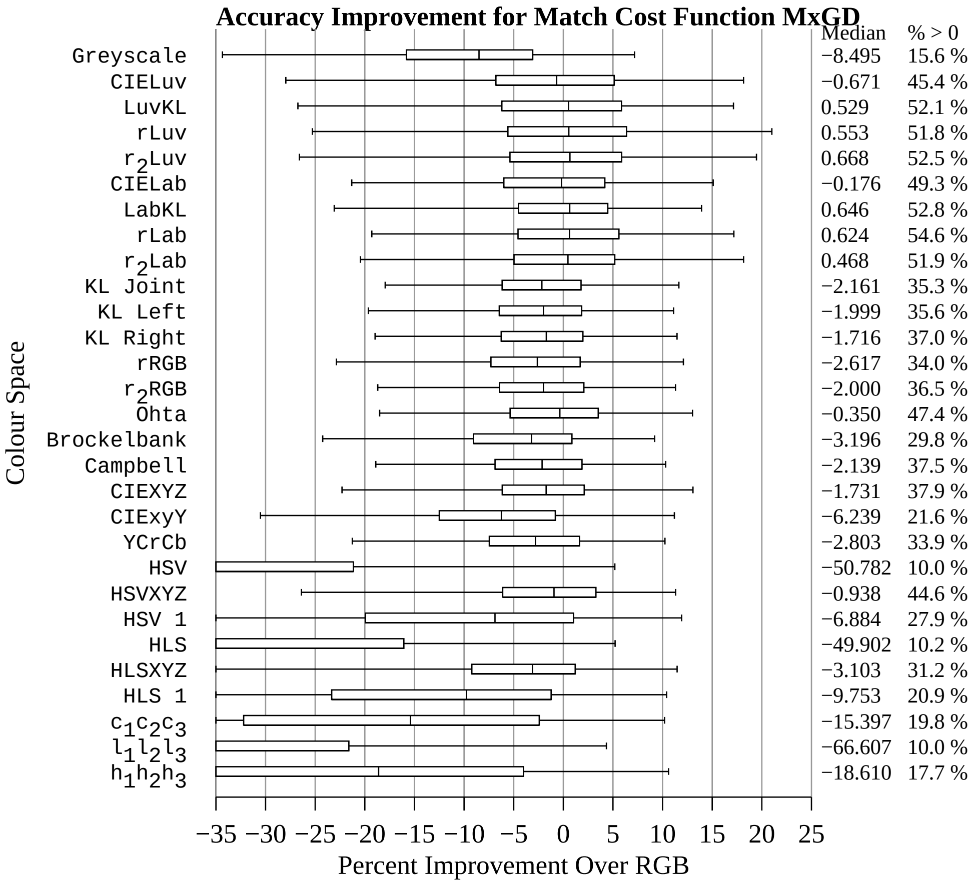


Figure J.2: Box plots of the over all aggregate percent accuracy improvement distributions for the MxGD match cost function for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

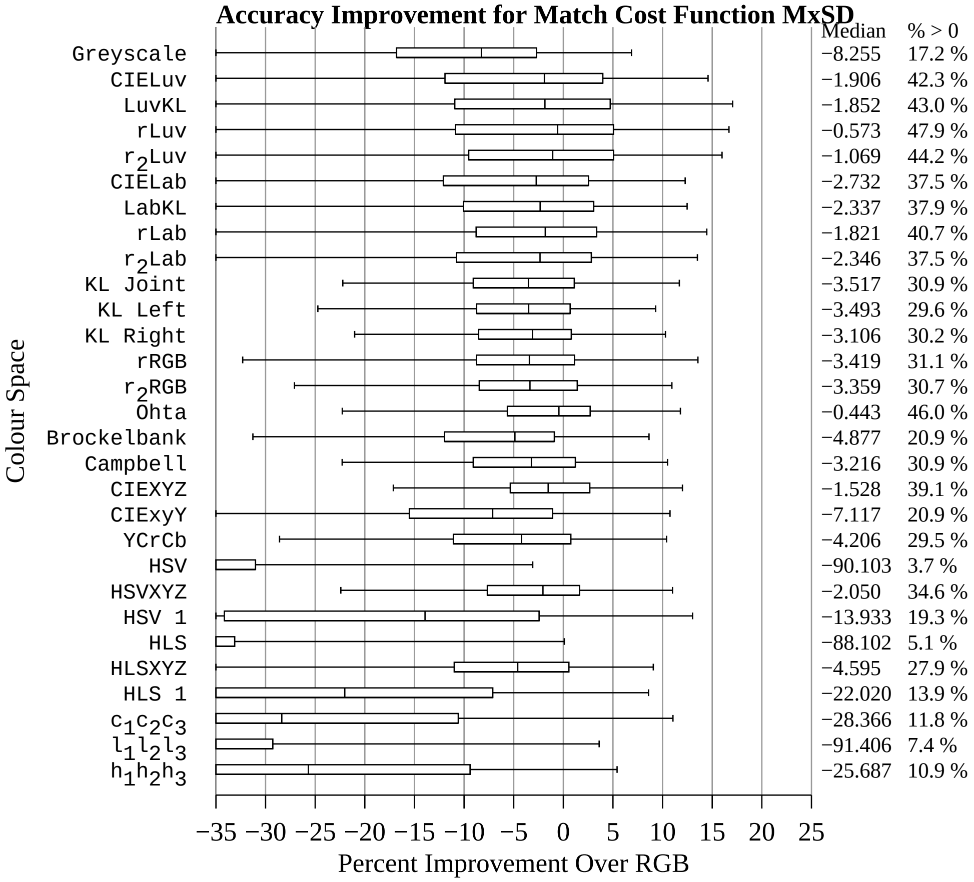


Figure J.3: Box plots of the over all aggregate percent accuracy improvement distributions for the MxSD match cost function for each colour representation. The plots indicate the fifth, 25th, 50th, 75th, and 95th percentiles of the distribution.

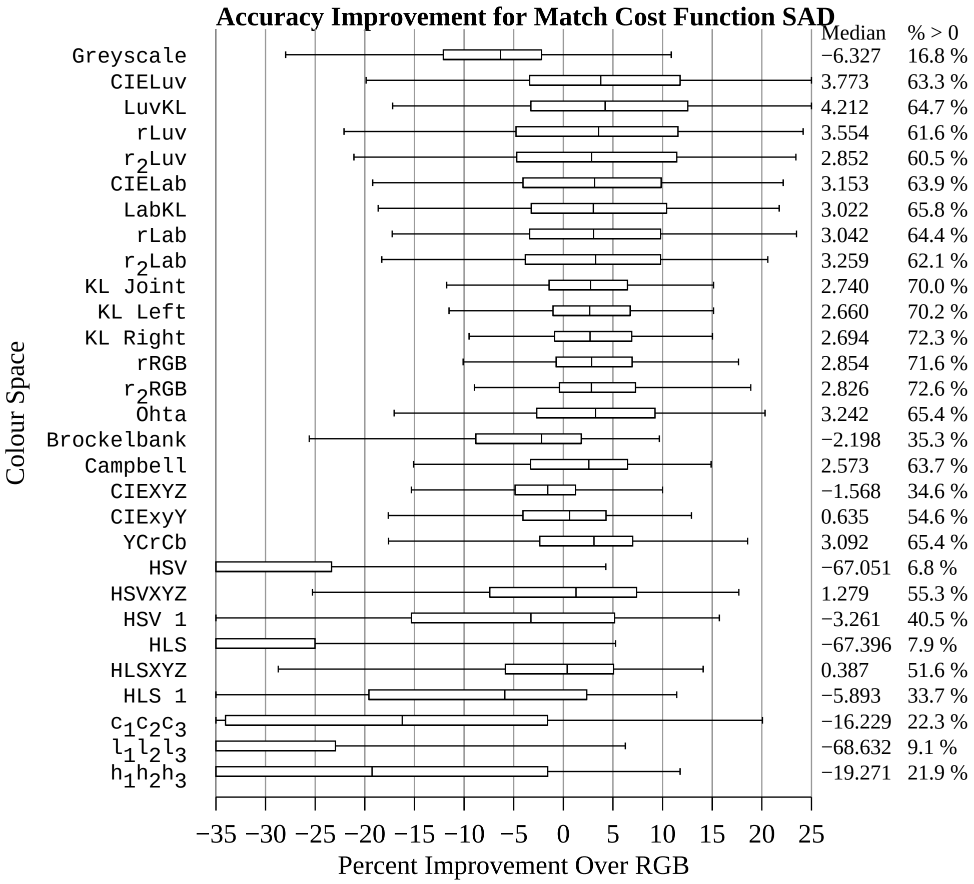


Figure J.4: Box plots of the over all aggregate percent accuracy improvement distributions for the SAD match cost function for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

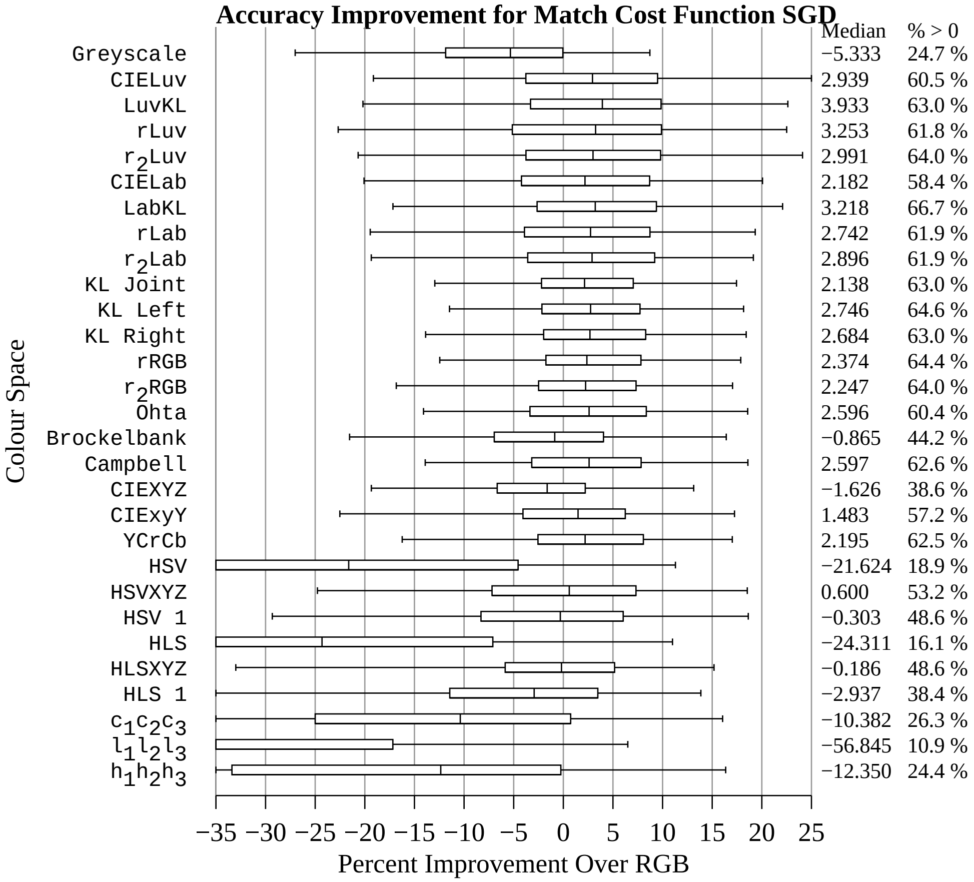


Figure J.5: Box plots of the over all aggregate percent accuracy improvement distributions for the SGD match cost function for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

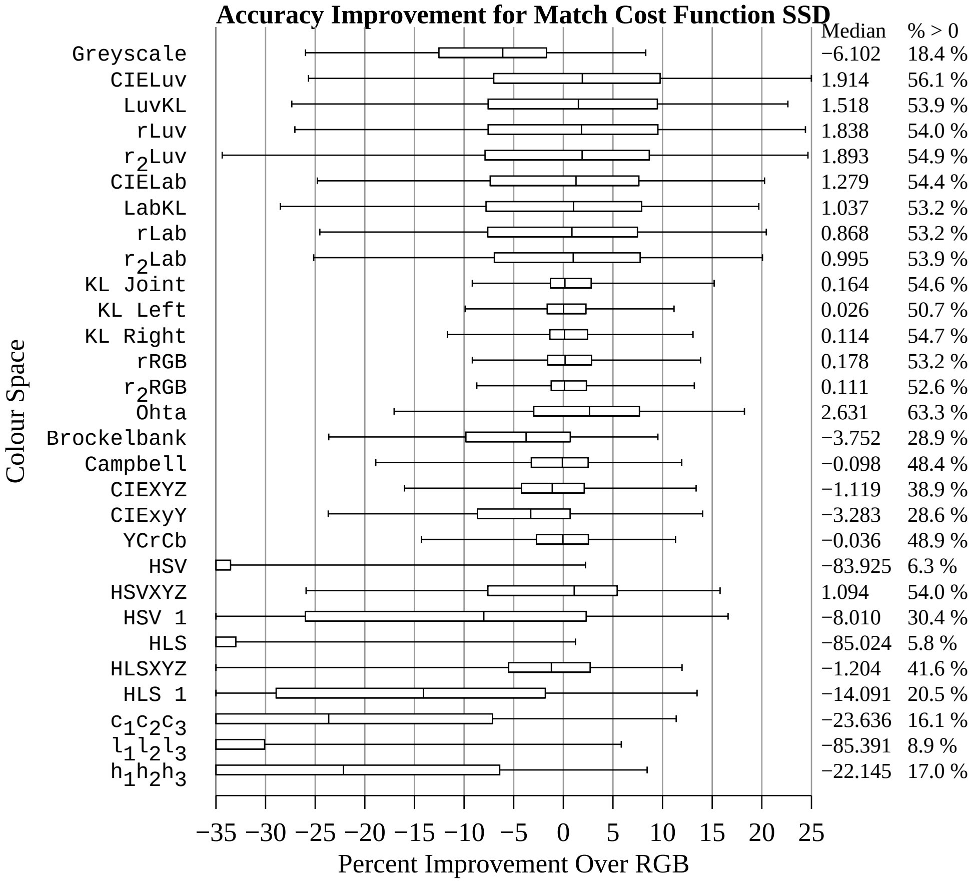


Figure J.6: Box plots of the over all aggregate percent accuracy improvement distributions for the SSD match cost function for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

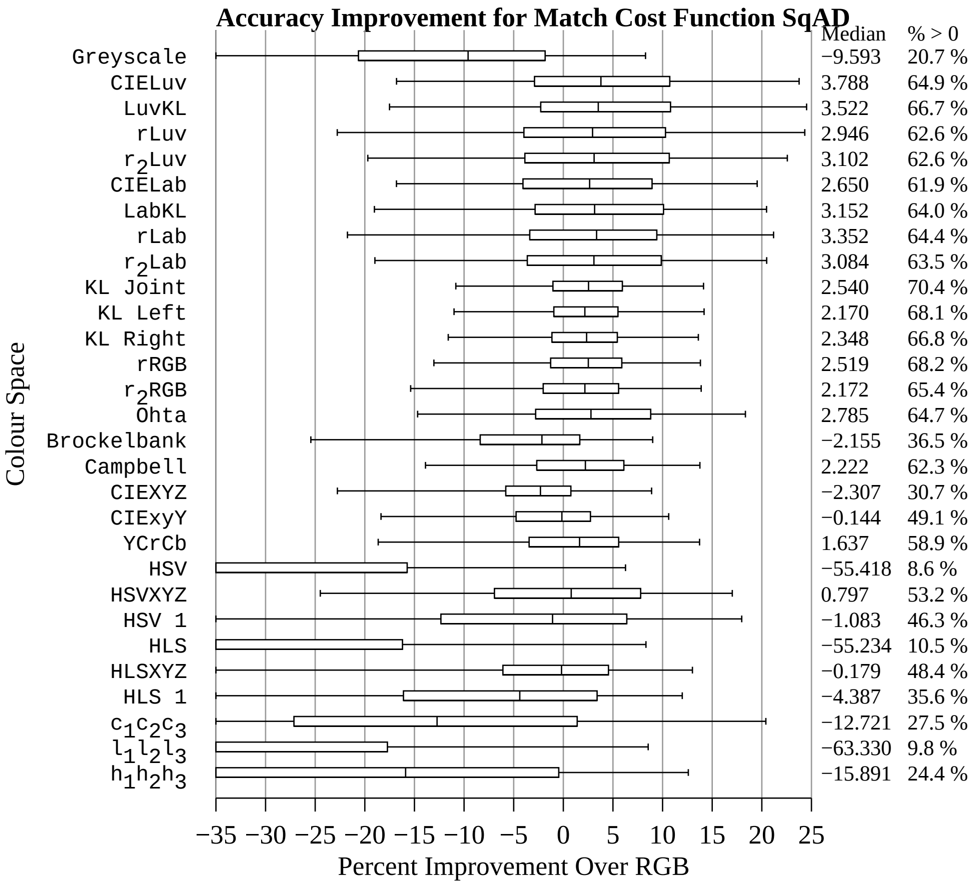


Figure J.7: Box plots of the over all aggregate percent accuracy improvement distributions for the SqAD match cost function for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

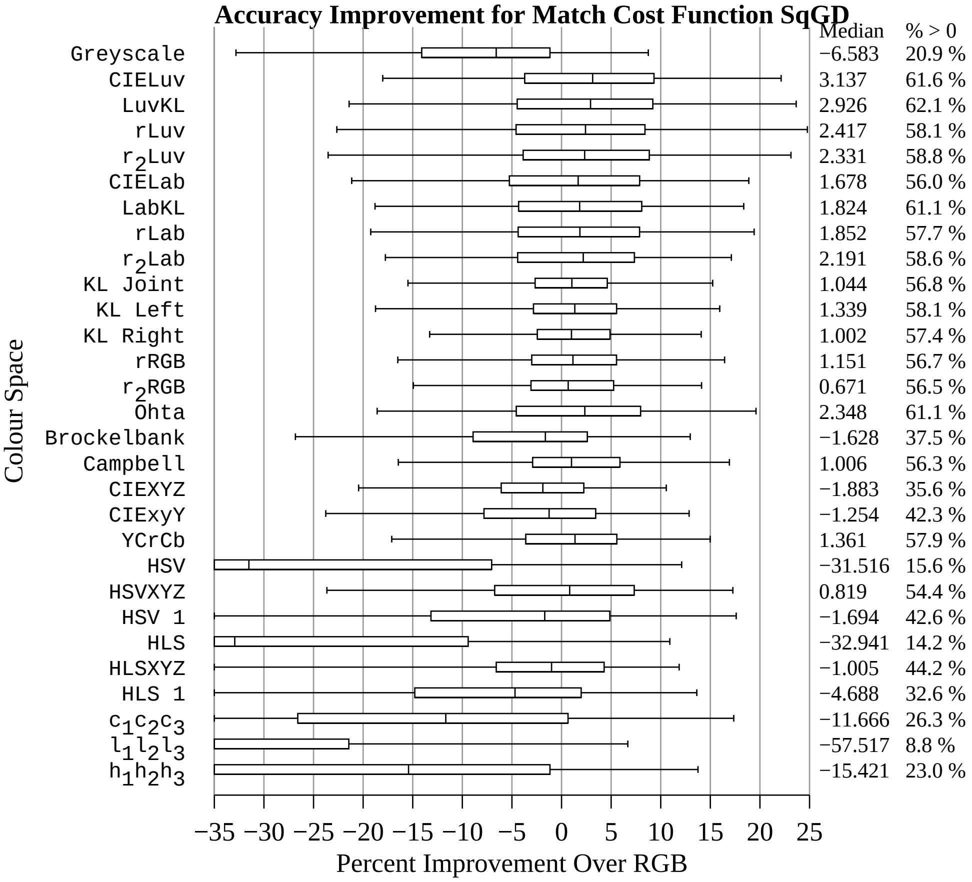


Figure J.8: Box plots of the over all aggregate percent accuracy improvement distributions for the SqGD match cost function for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

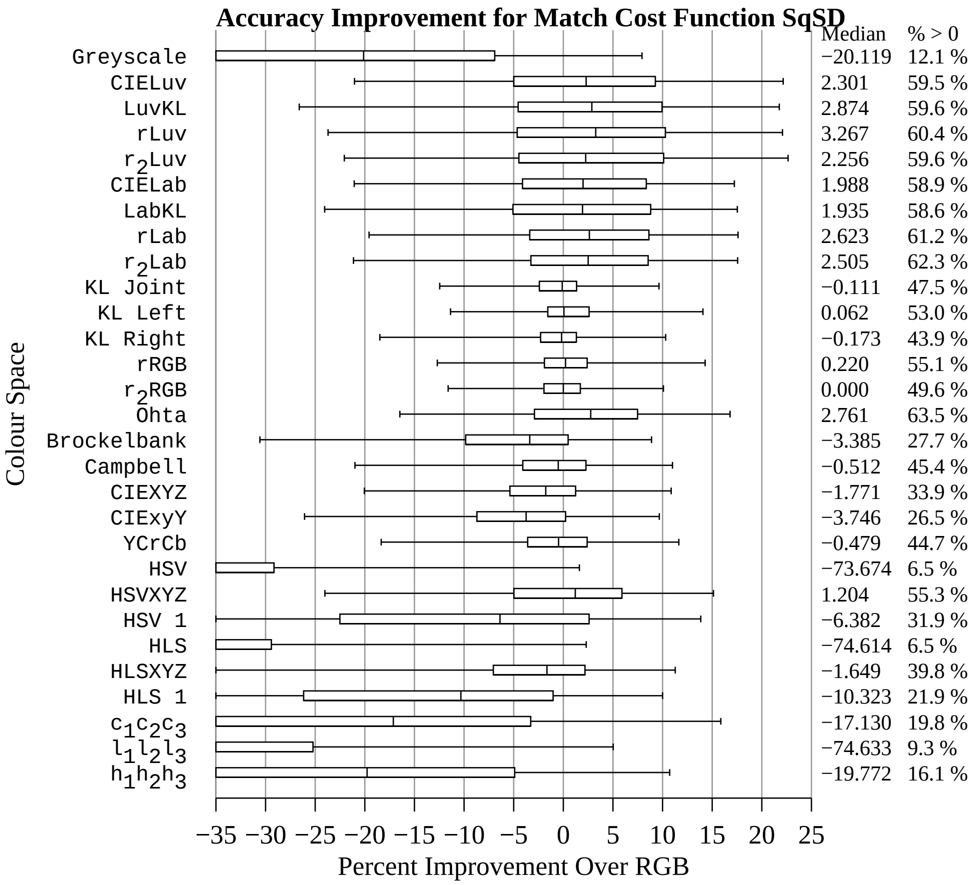


Figure J.9: Box plots of the over all aggregate percent accuracy improvement distributions for the SqSD match cost function for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

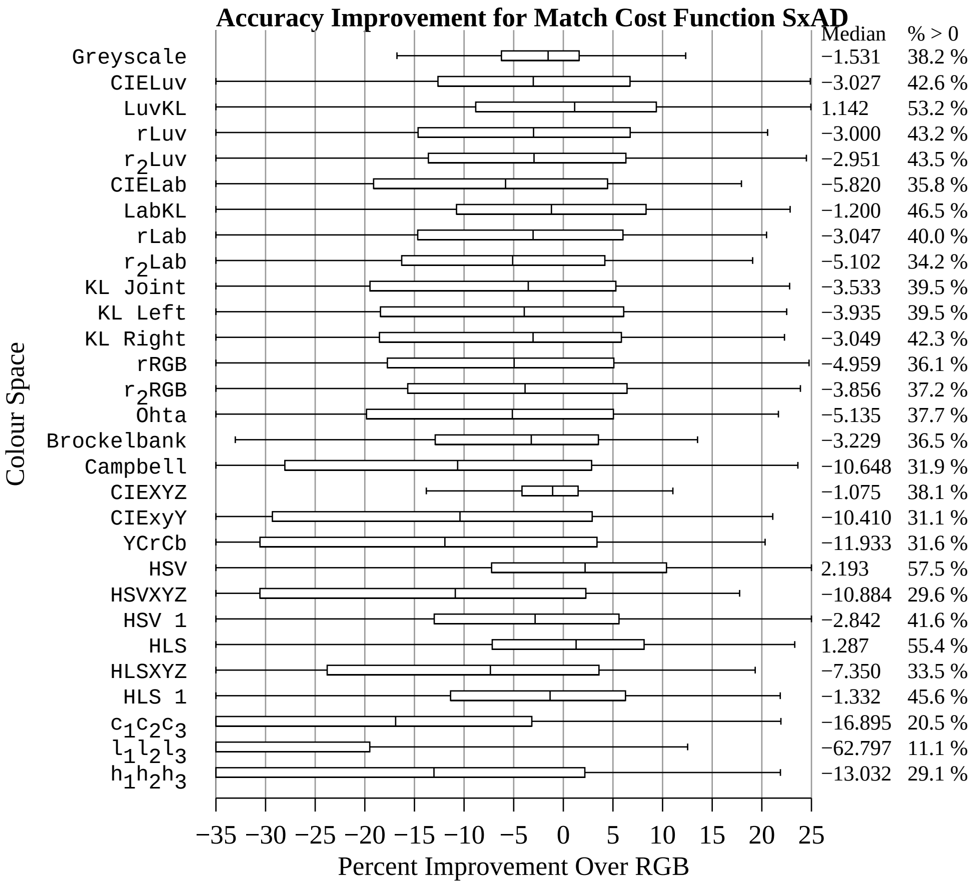


Figure J.10: Box plots of the over all aggregate percent accuracy improvement distributions for the SxAD match cost function for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

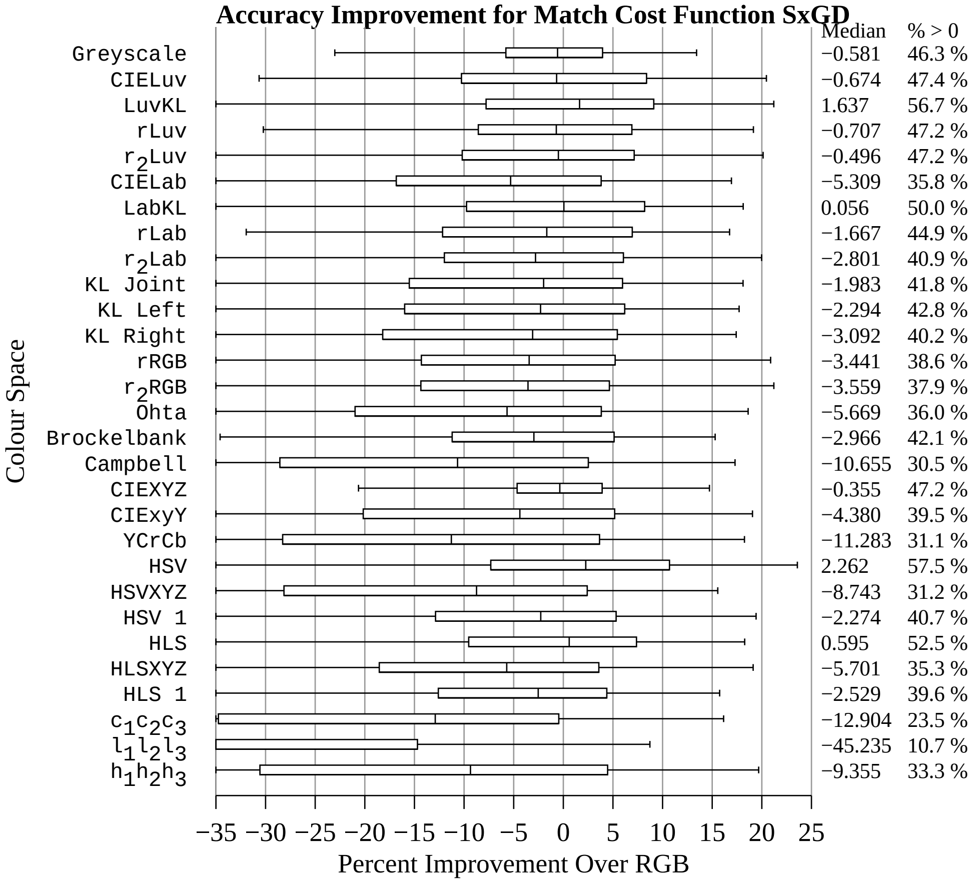


Figure J.11: Box plots of the over all aggregate percent accuracy improvement distributions for the SxGD match cost function for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

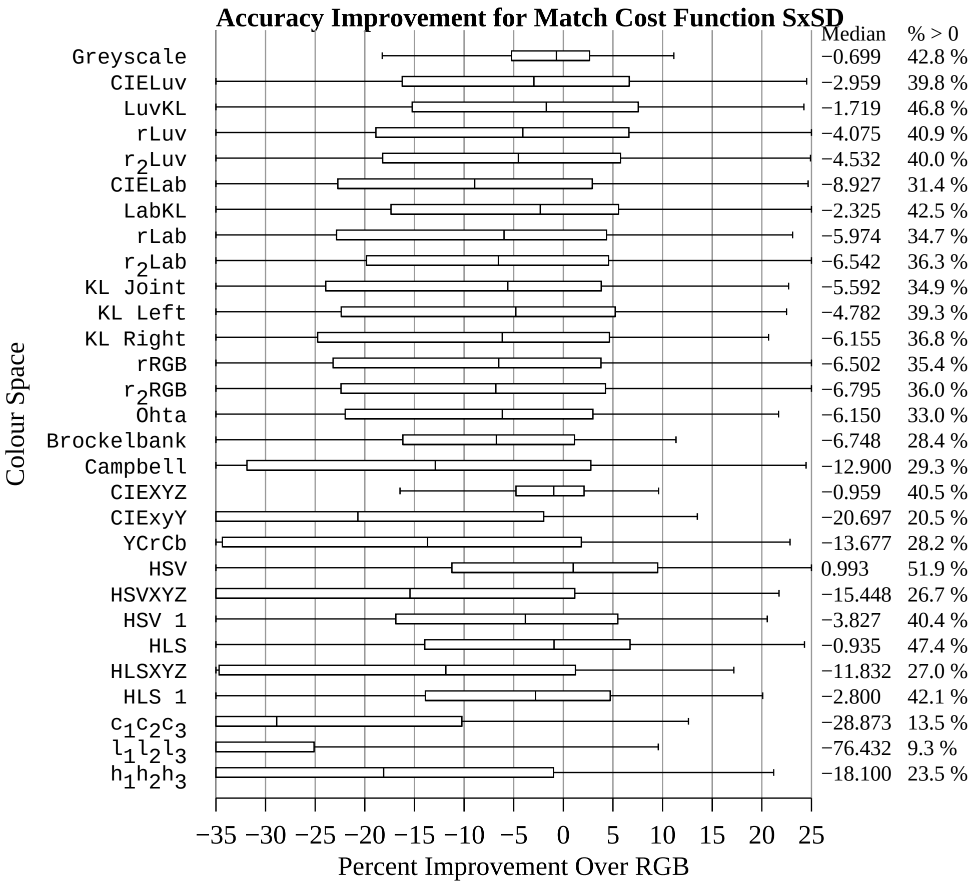


Figure J.12: Box plots of the over all aggregate percent accuracy improvement distributions for the SxSD match cost function for each colour representation. The plots indicate the fifth, 25th, 50th, 75th, and 95th percentiles of the distribution.

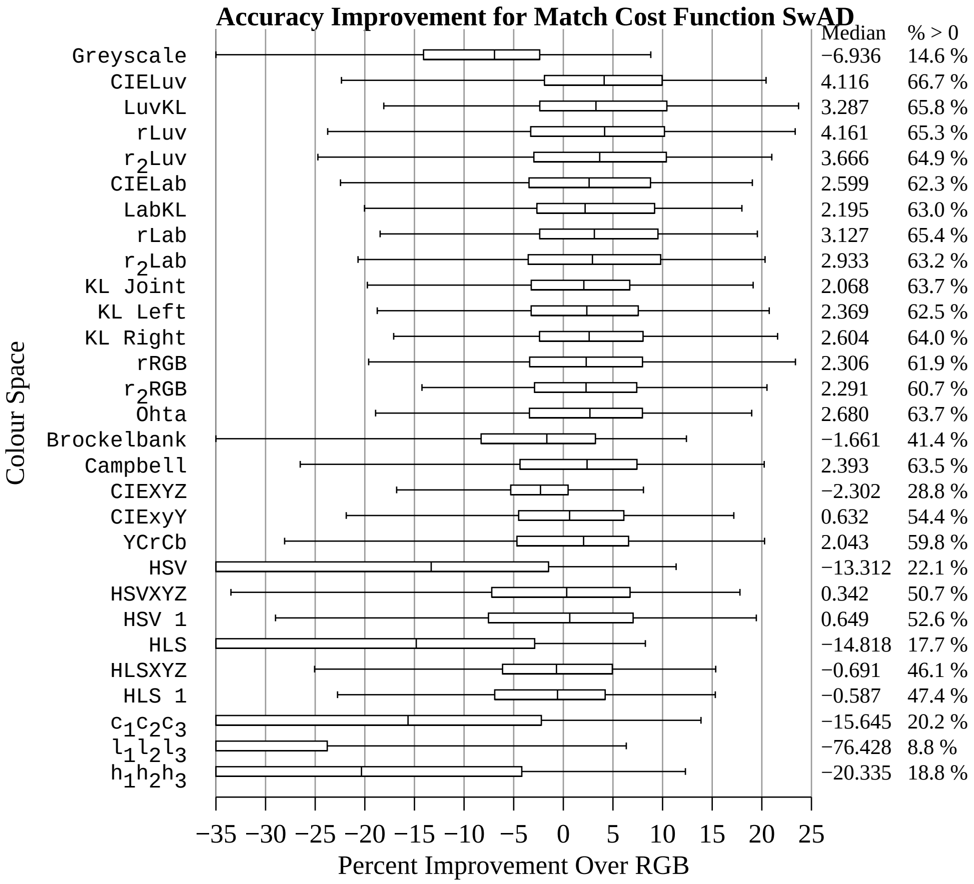


Figure J.13: Box plots of the over all aggregate percent accuracy improvement distributions for the SwAD match cost function for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

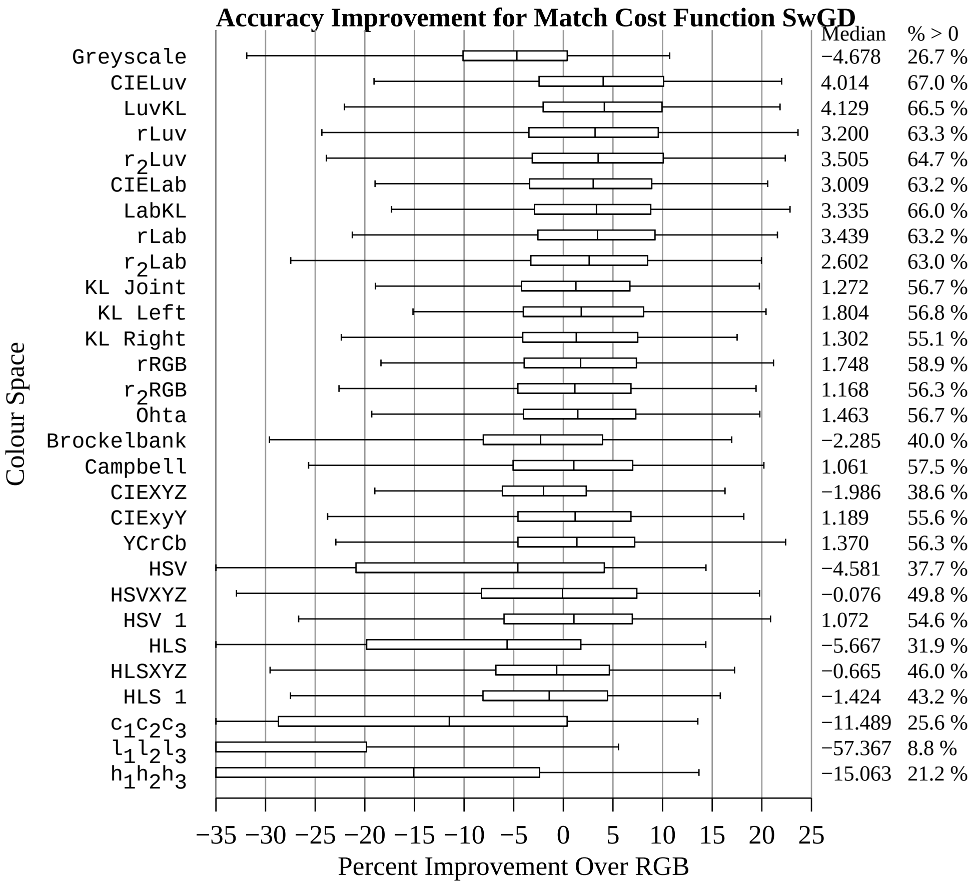


Figure J.14: Box plots of the over all aggregate percent accuracy improvement distributions for the SwGD match cost function for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

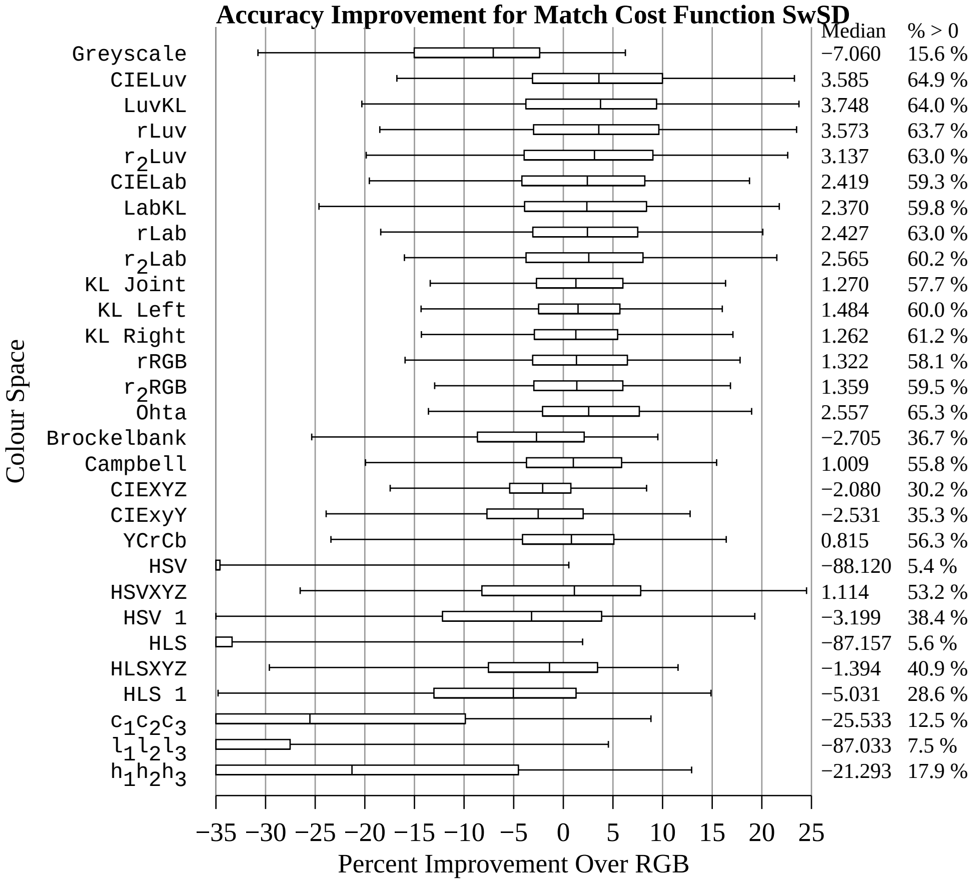


Figure J.15: Box plots of the over all aggregate percent accuracy improvement distributions for the SwSD match cost function for each colour representation. The plots indicate the fifth, 25th, 50th, 75th, and 95th percentiles of the distribution.

Appendix K

Aggregate Improvement for Colour Spaces by Match Cost Function

K.1 SO1 Framework Algorithms

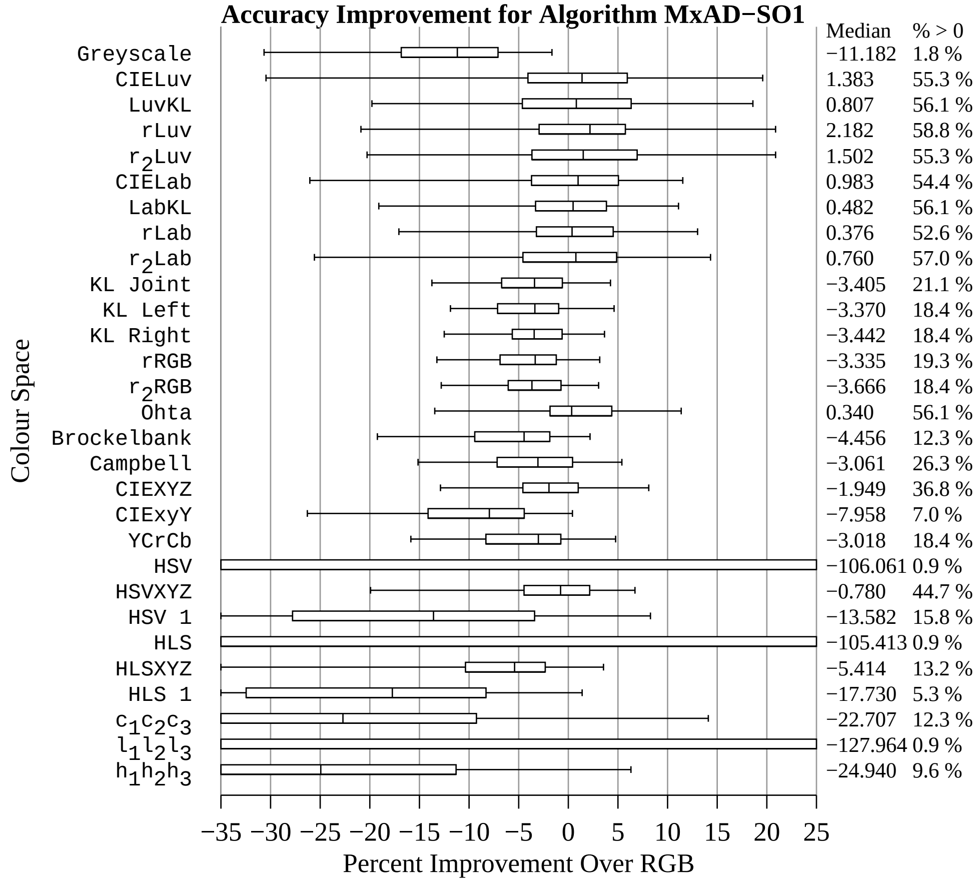


Figure K.1: Box plots of the over all aggregate percent accuracy improvement distributions for the MxAD-SO1 algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

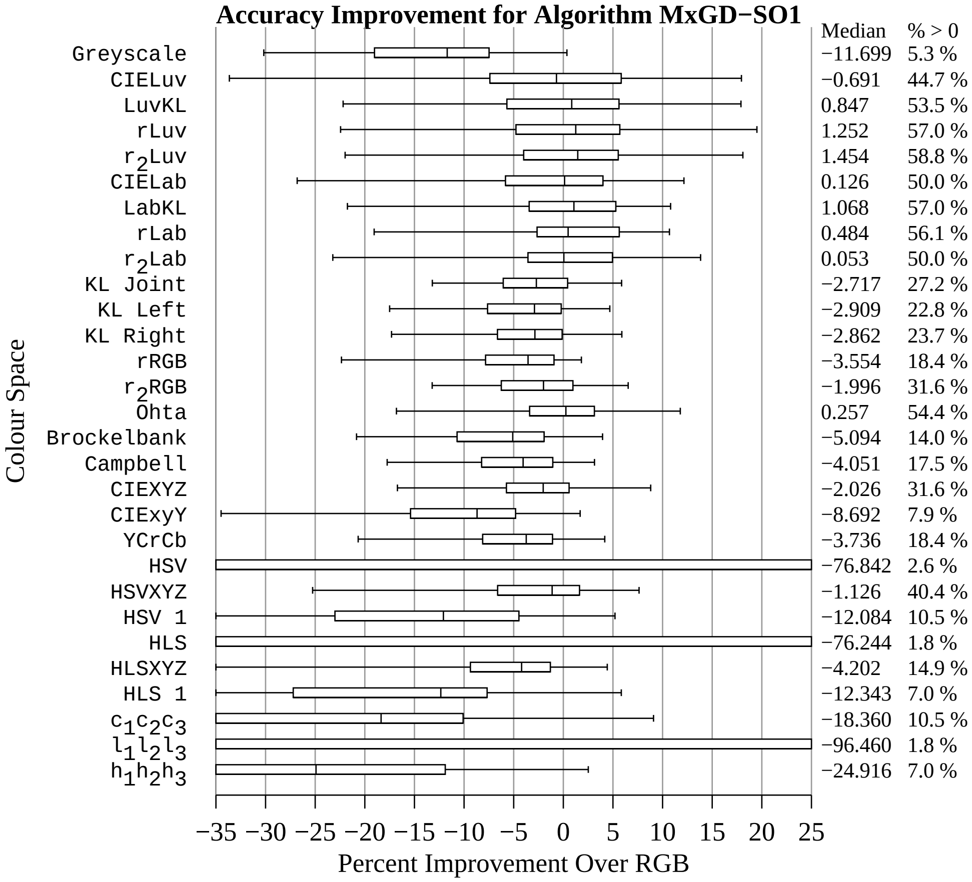


Figure K.2: Box plots of the over all aggregate percent accuracy improvement distributions for the MxGD-SO1 algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

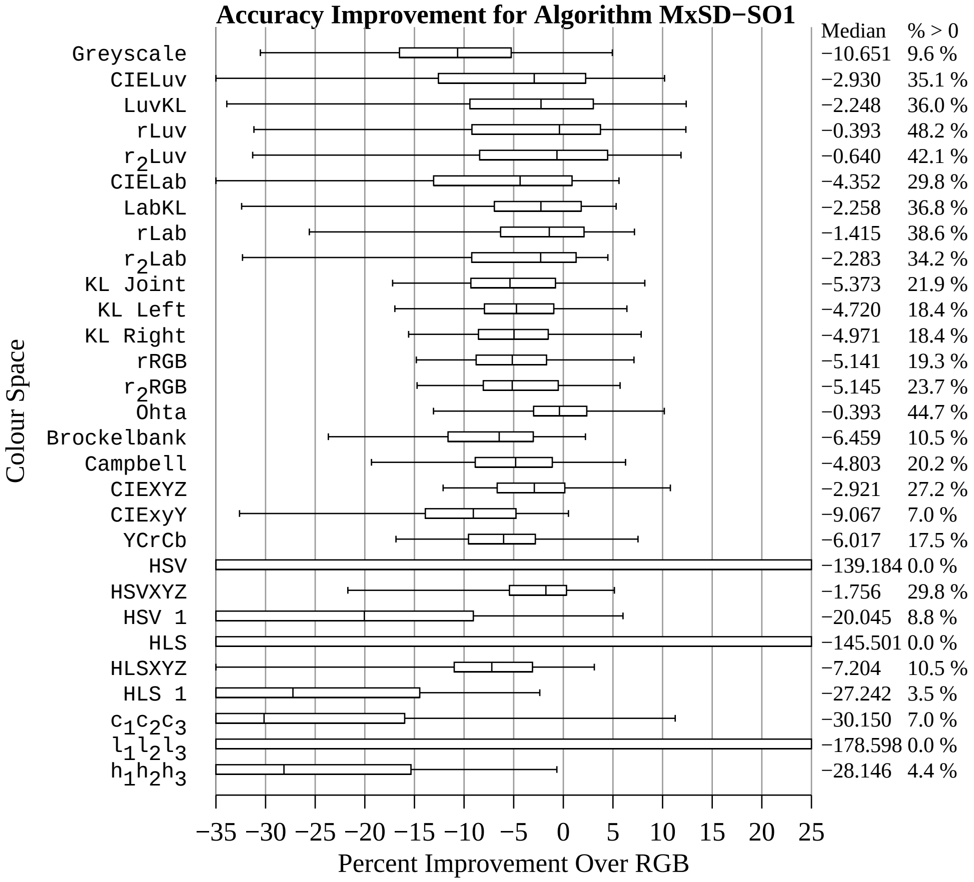


Figure K.3: Box plots of the over all aggregate percent accuracy improvement distributions for the MxSD-SO1 algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

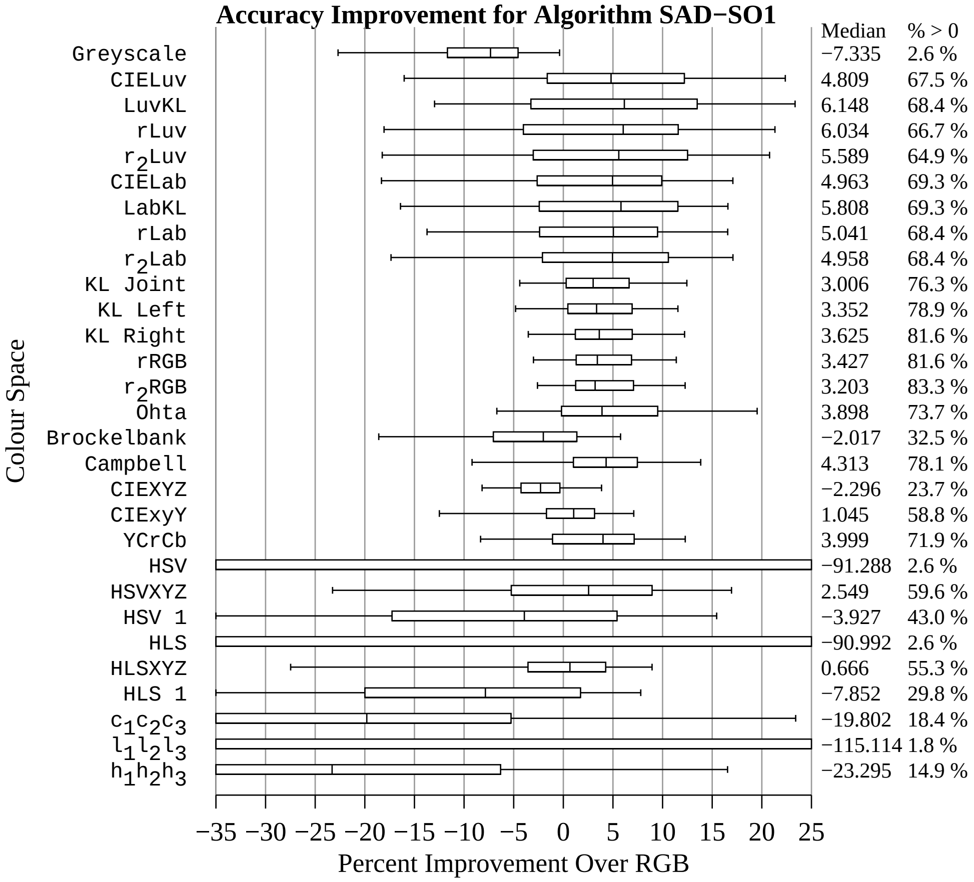


Figure K.4: Box plots of the over all aggregate percent accuracy improvement distributions for the SAD-SO1 algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

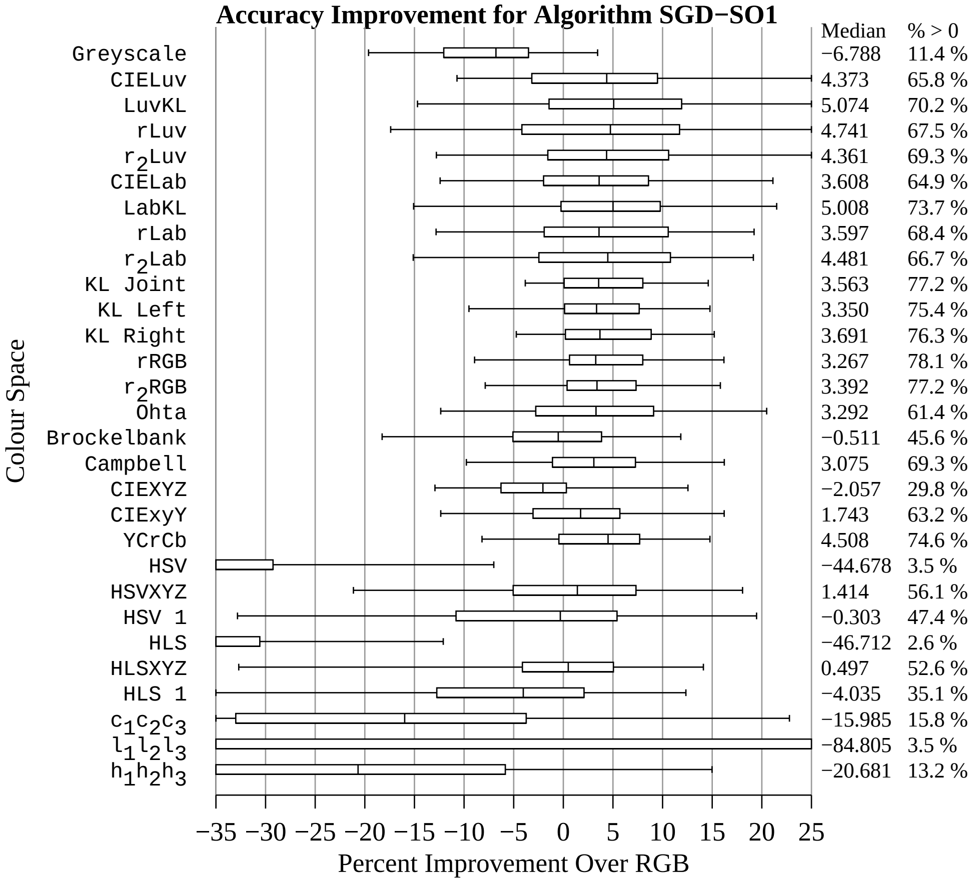


Figure K.5: Box plots of the over all aggregate percent accuracy improvement distributions for the SGD-SO1 algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

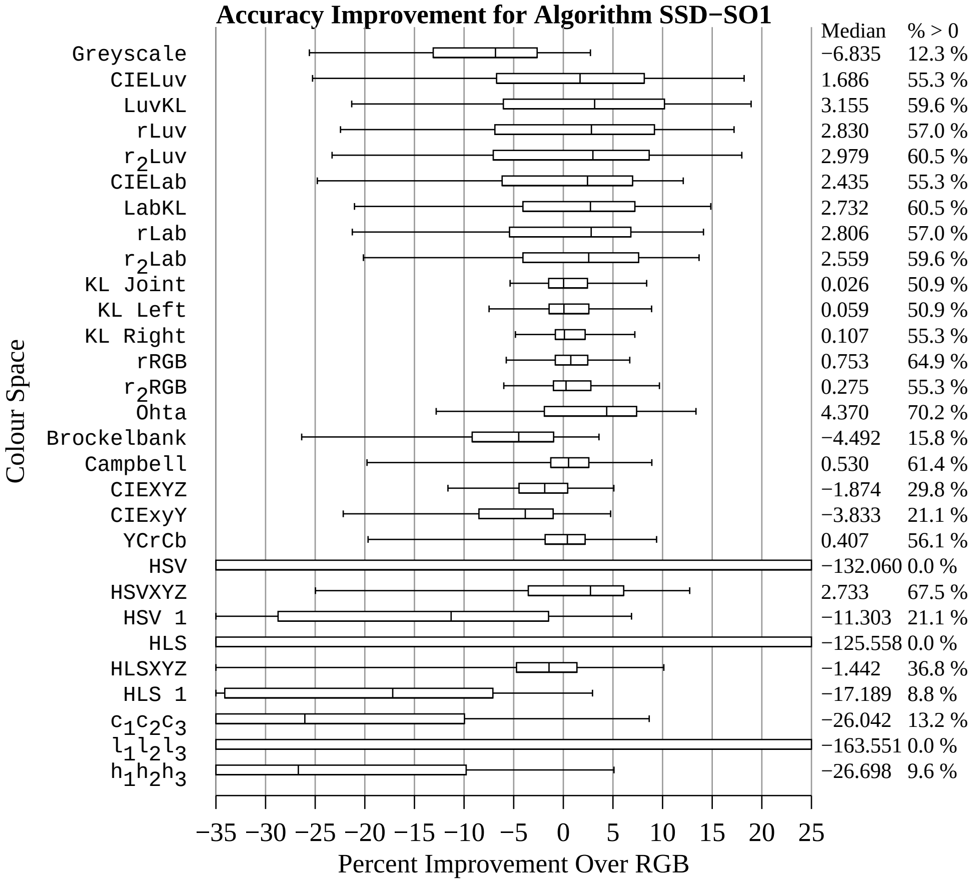


Figure K.6: Box plots of the over all aggregate percent accuracy improvement distributions for the SSD-SO1 algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

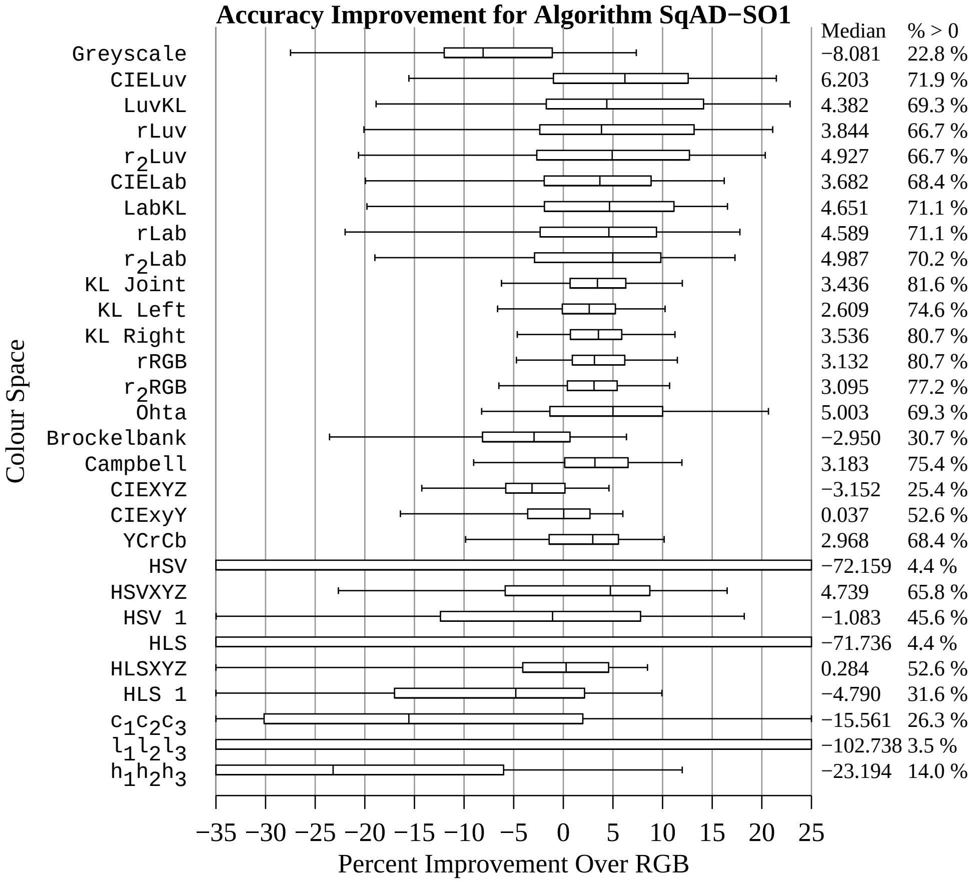


Figure K.7: Box plots of the over all aggregate percent accuracy improvement distributions for the SqAD-SO1 algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

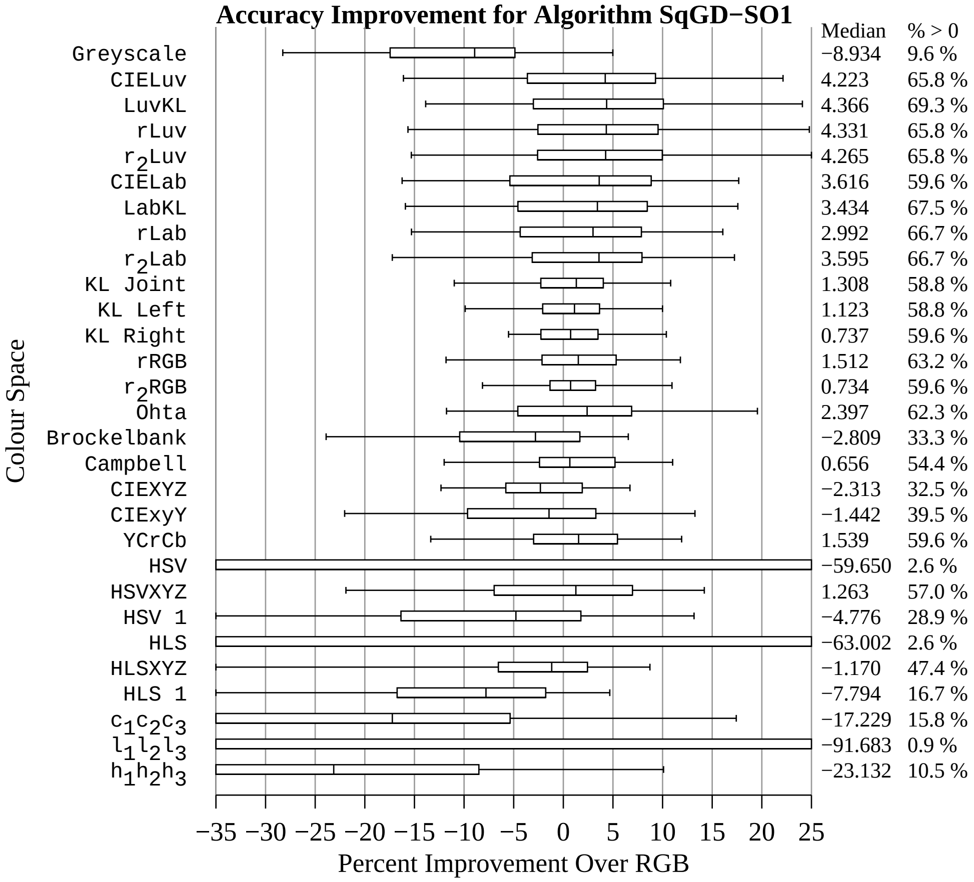


Figure K.8: Box plots of the over all aggregate percent accuracy improvement distributions for the SqGD-SO1 algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

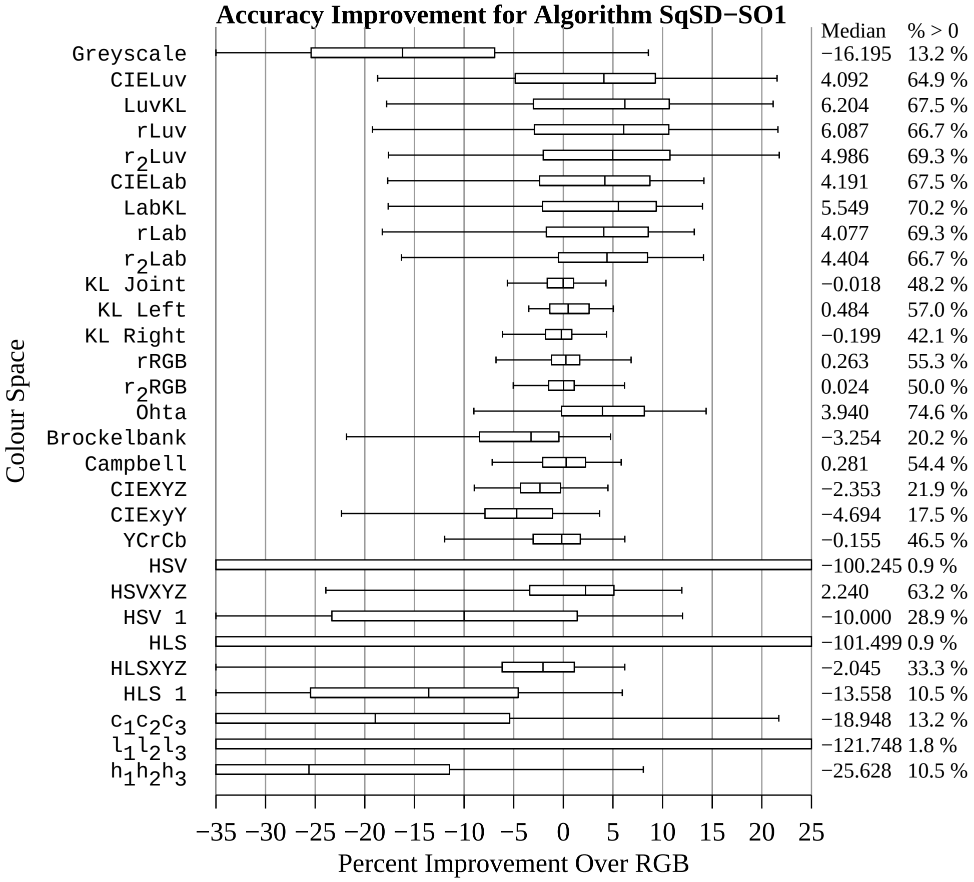


Figure K.9: Box plots of the over all aggregate percent accuracy improvement distributions for the SqSD-SO1 algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

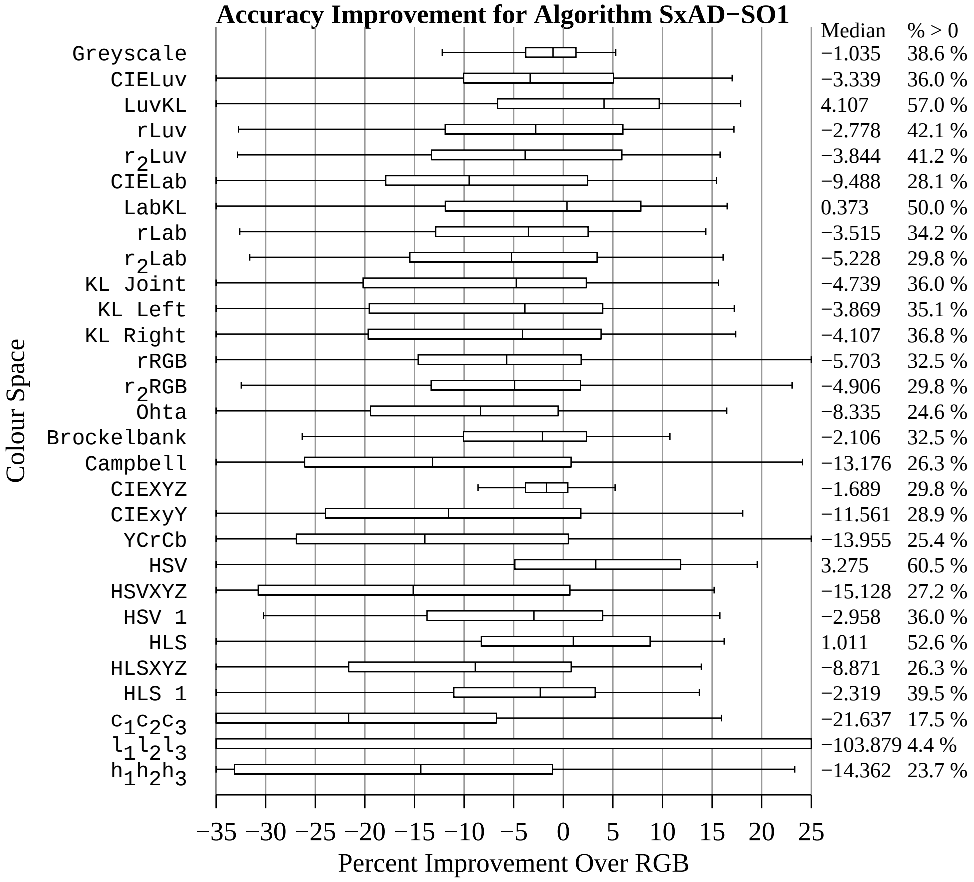


Figure K.10: Box plots of the over all aggregate percent accuracy improvement distributions for the SxAD-SO1 algorithm for each colour representation. The plots indicate the fifth, 25th, 50th, 75th, and 95th percentiles of the distribution.

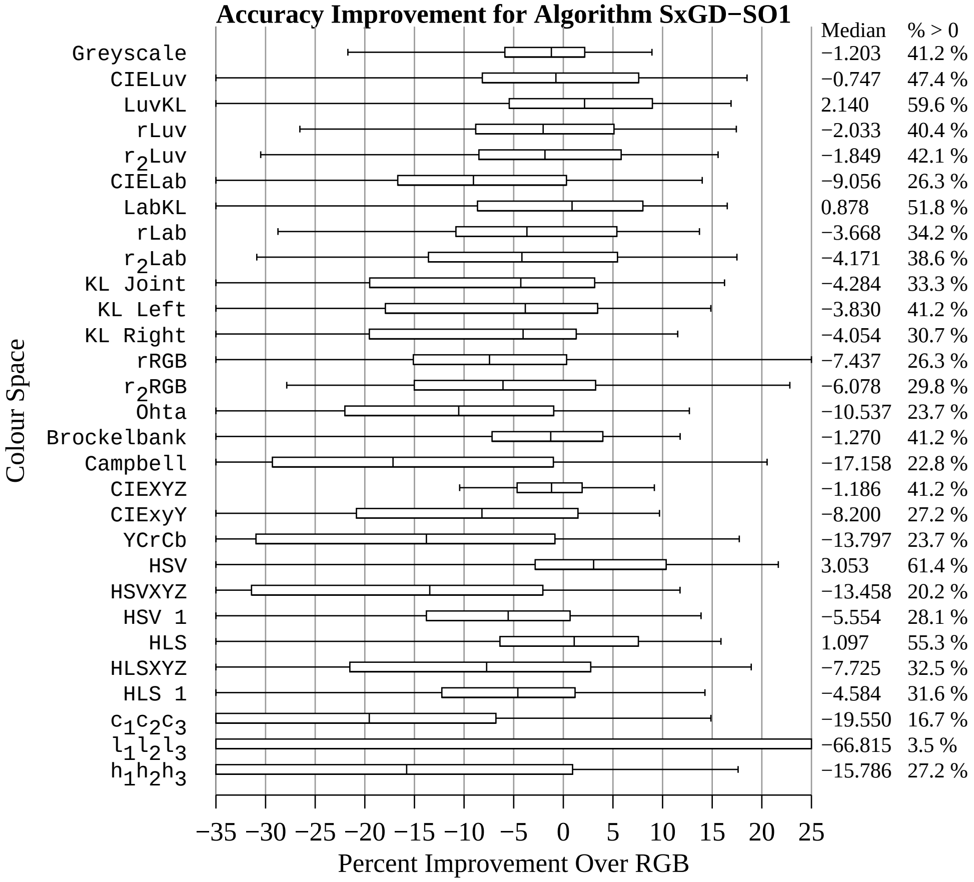


Figure K.11: Box plots of the over all aggregate percent accuracy improvement distributions for the SxGD-SO1 algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

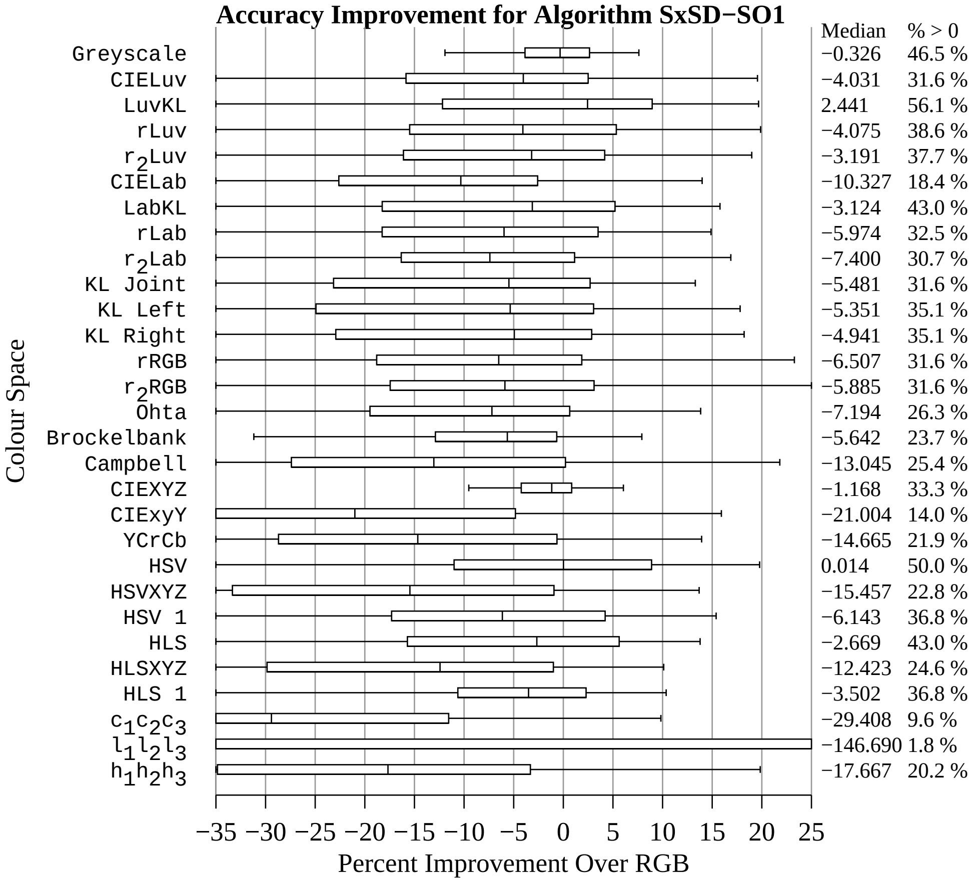


Figure K.12: Box plots of the over all aggregate percent accuracy improvement distributions for the SxSD-SO1 algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

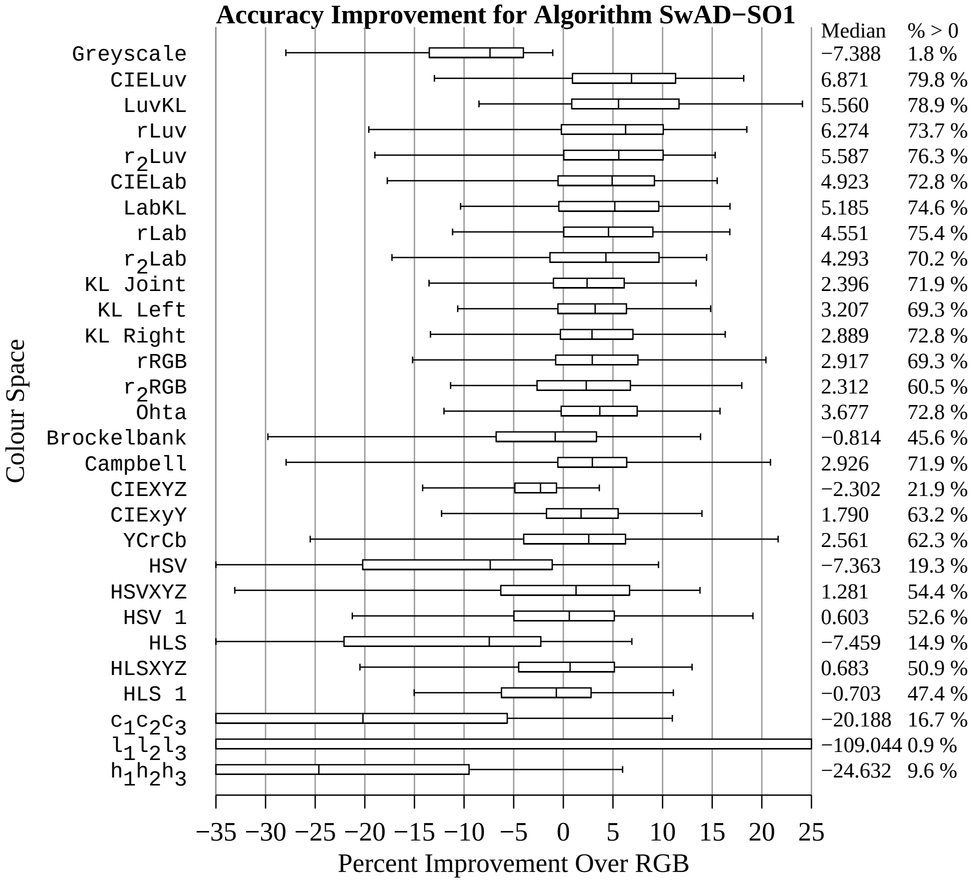


Figure K.13: Box plots of the over all aggregate percent accuracy improvement distributions for the SwAD-SO1 algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

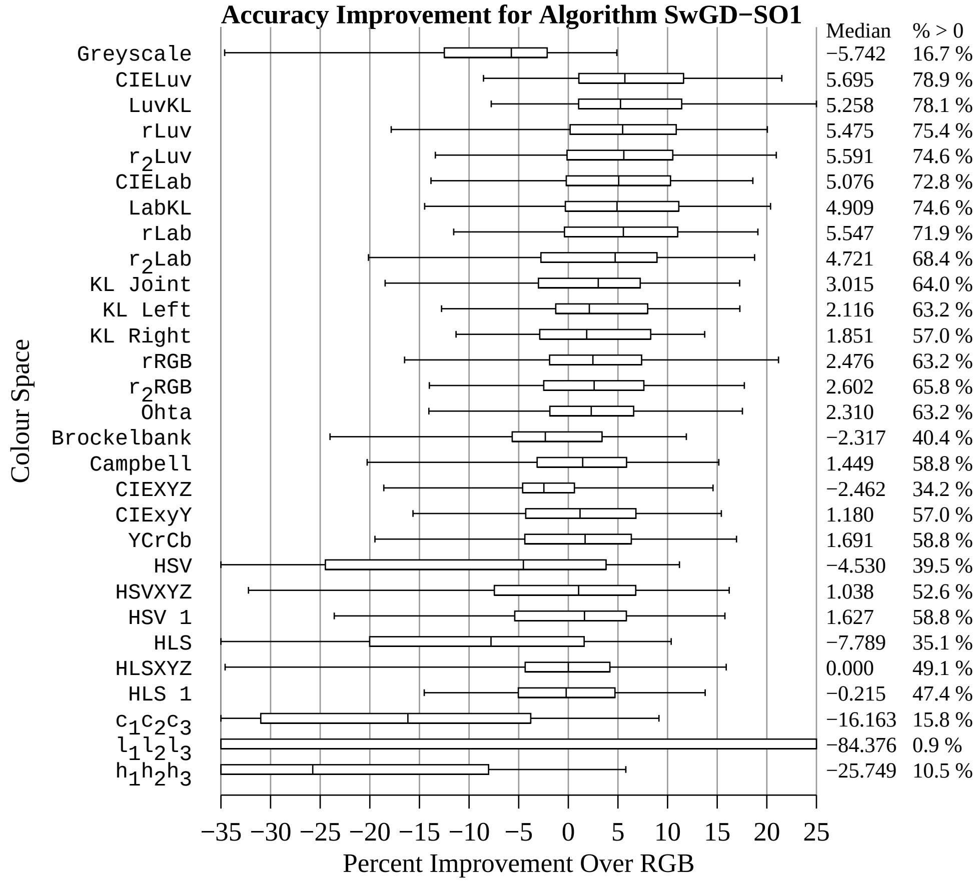


Figure K.14: Box plots of the over all aggregate percent accuracy improvement distributions for the SwGD-SO1 algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

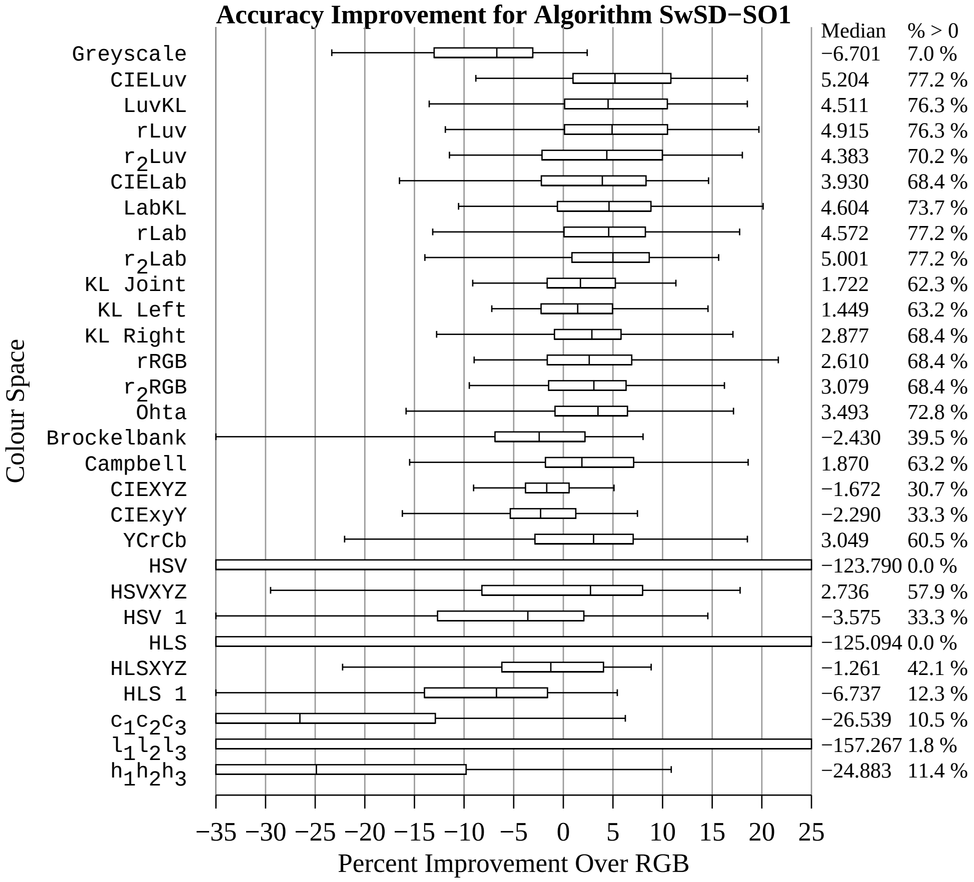


Figure K.15: Box plots of the over all aggregate percent accuracy improvement distributions for the SwSD-SO1 algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

K.2 SO2 Framework Algorithms

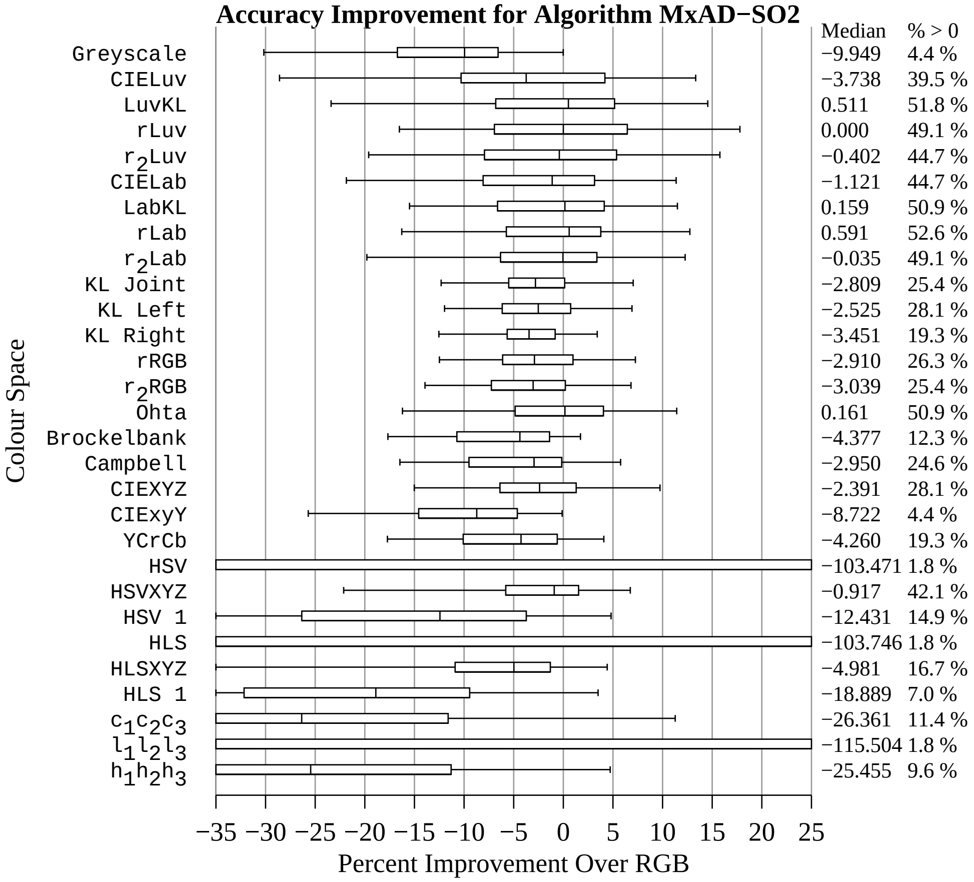


Figure K.16: Box plots of the over all aggregate percent accuracy improvement distributions for the MxAD-SO2 algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

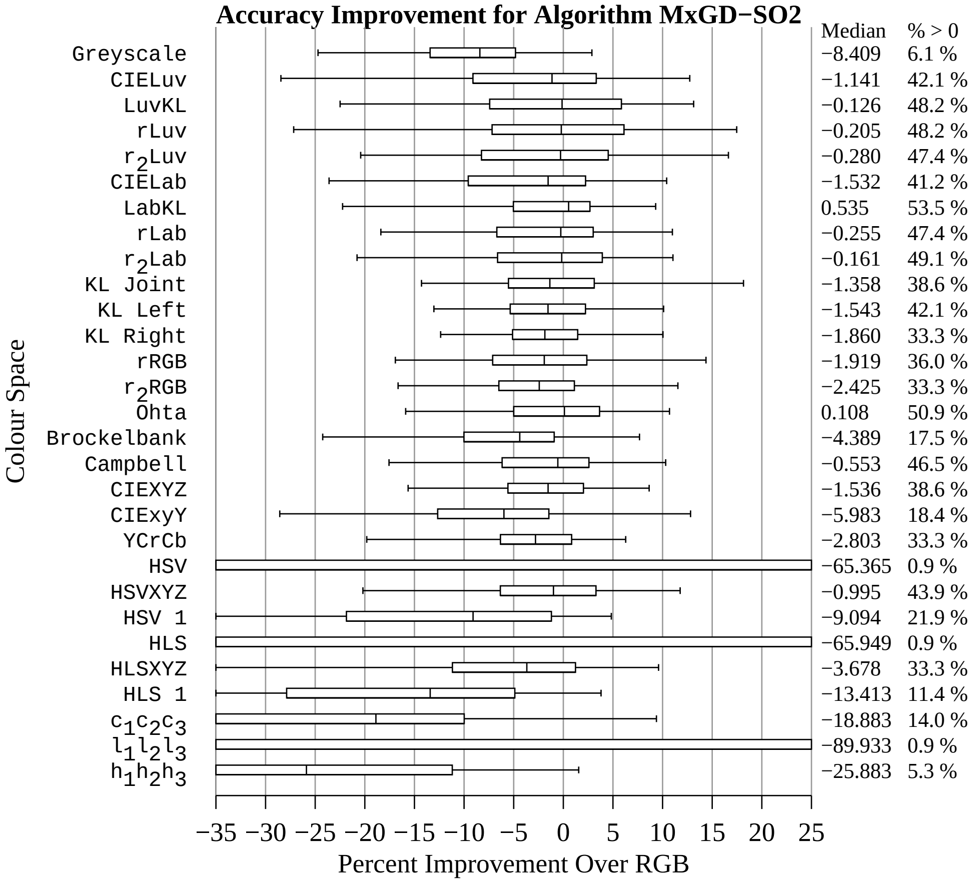


Figure K.17: Box plots of the over all aggregate percent accuracy improvement distributions for the MxGD-SO2 algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

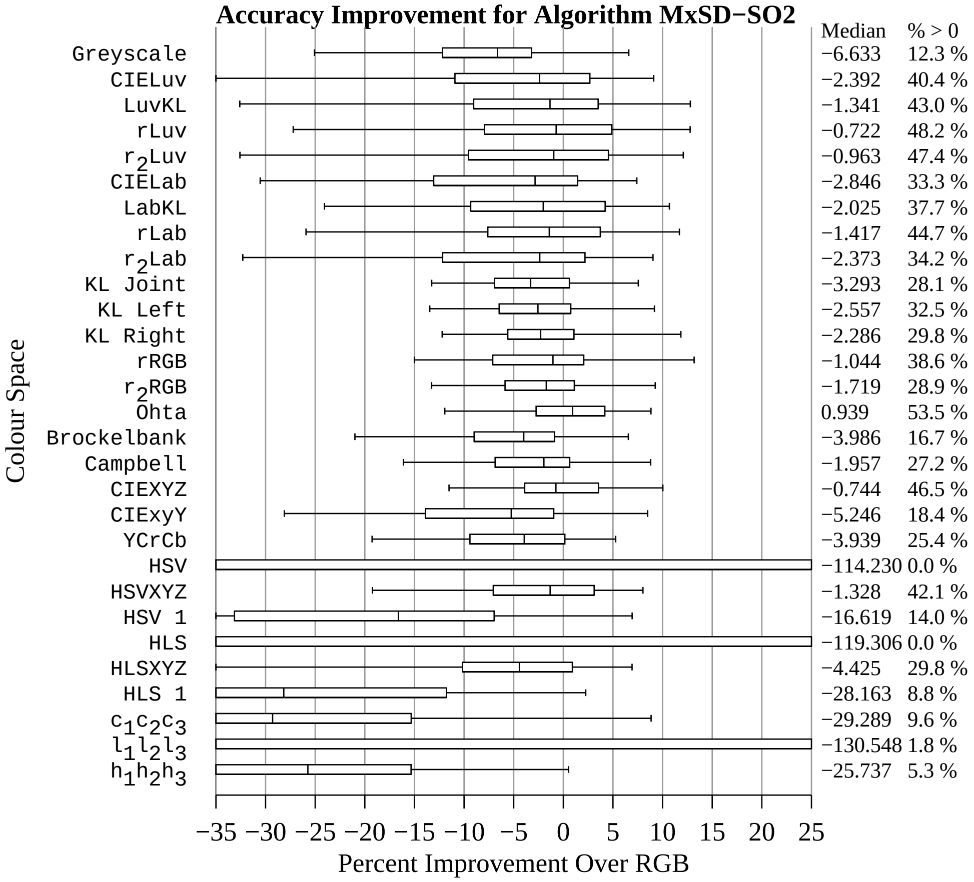


Figure K.18: Box plots of the over all aggregate percent accuracy improvement distributions for the MxSD-SO2 algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

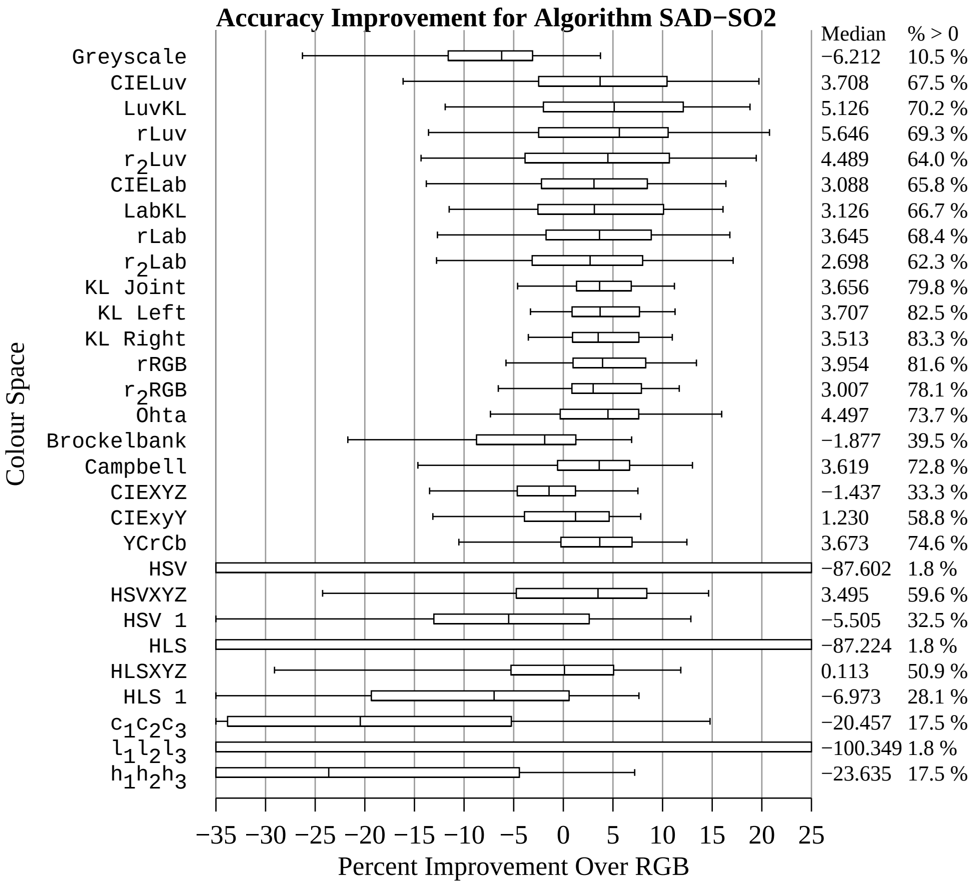


Figure K.19: Box plots of the over all aggregate percent accuracy improvement distributions for the SAD-SO2 algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

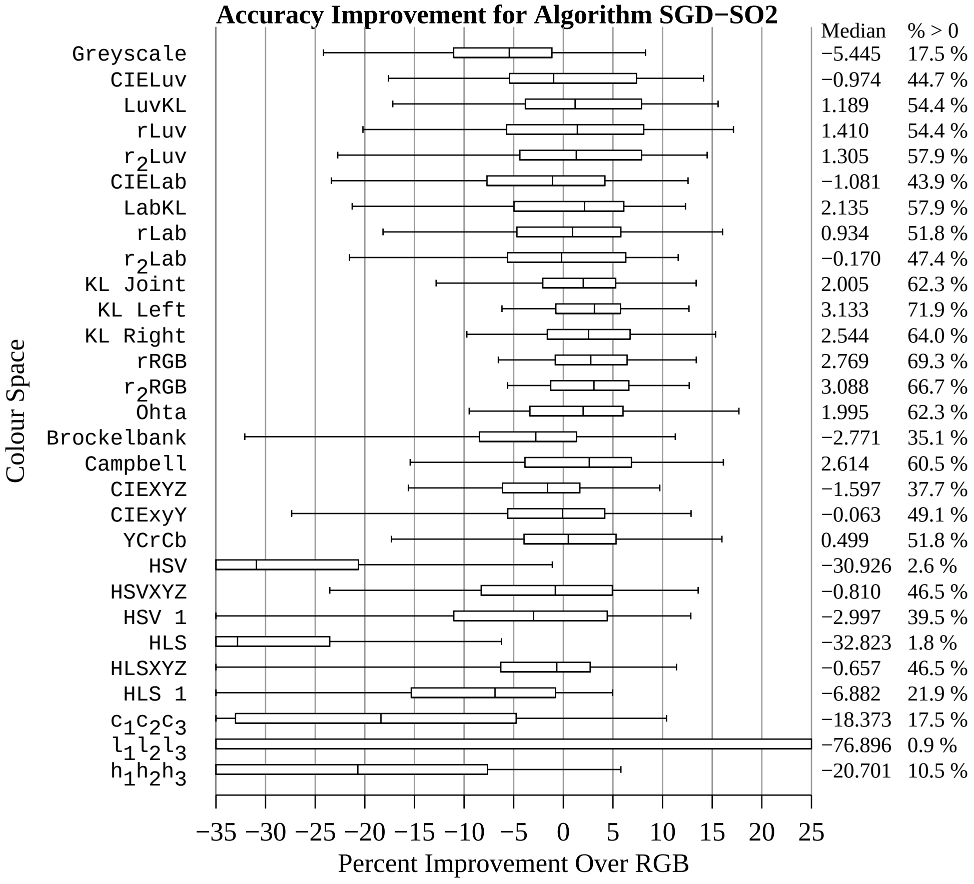


Figure K.20: Box plots of the over all aggregate percent accuracy improvement distributions for the SGD-SO2 algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

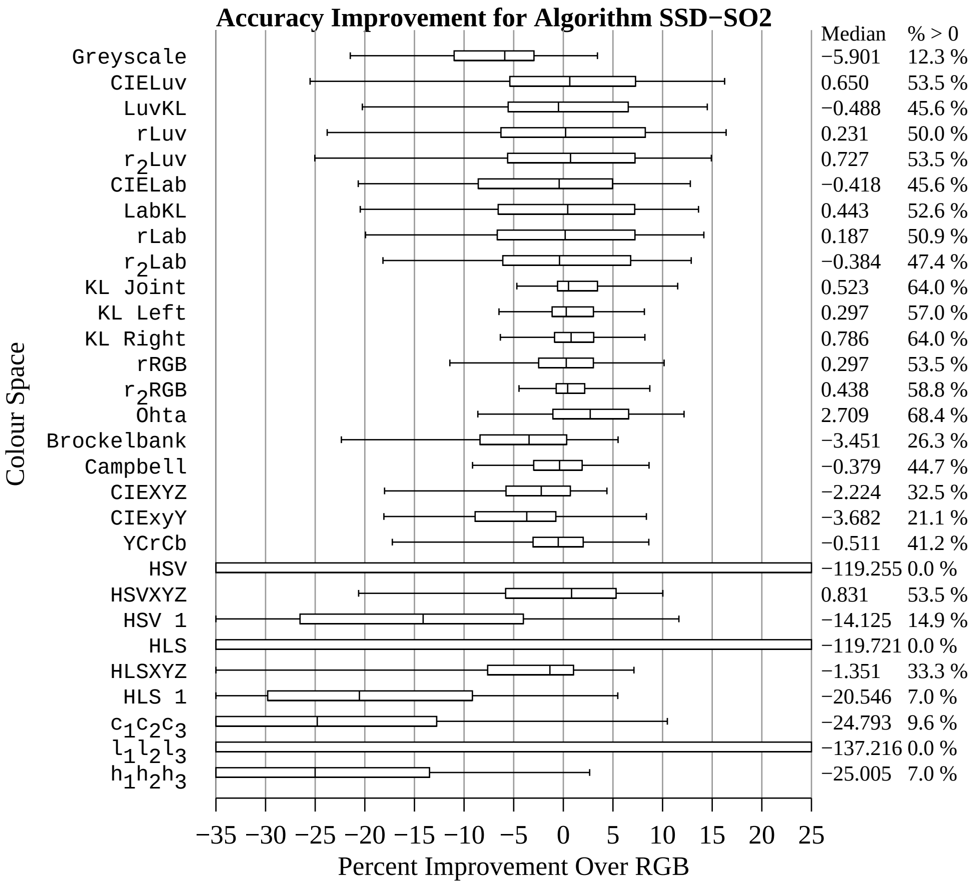


Figure K.21: Box plots of the over all aggregate percent accuracy improvement distributions for the SSD-SO2 algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

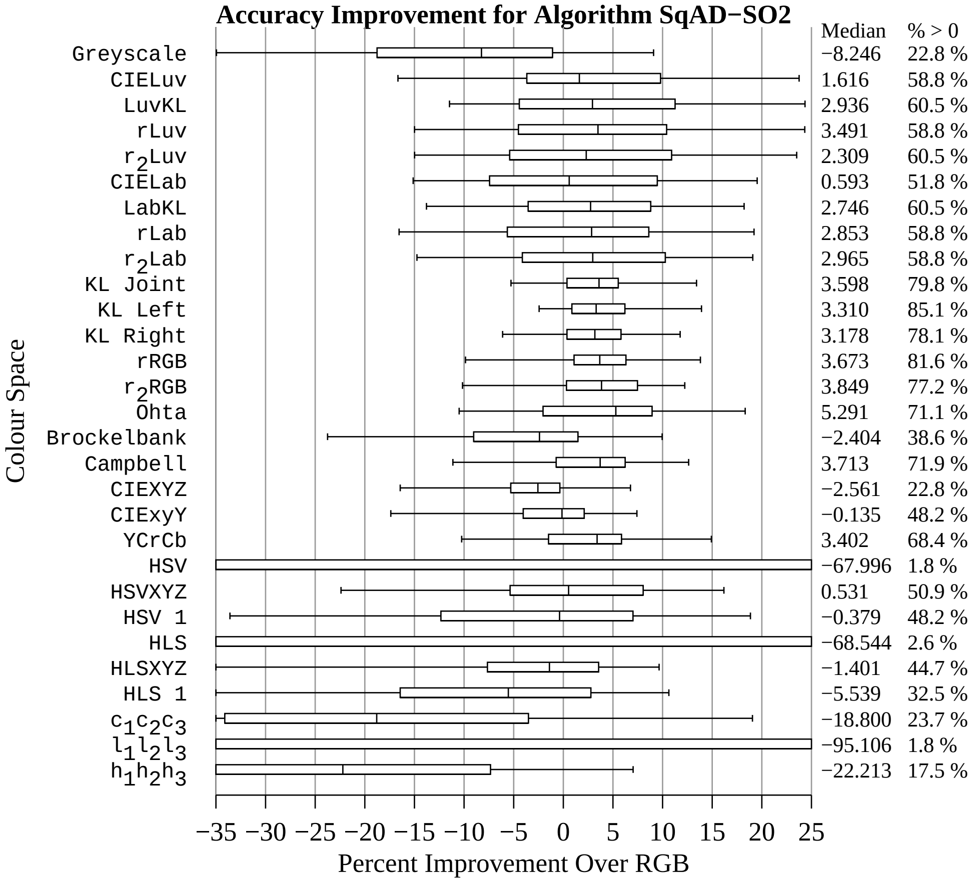


Figure K.22: Box plots of the over all aggregate percent accuracy improvement distributions for the SqAD-SO2 algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

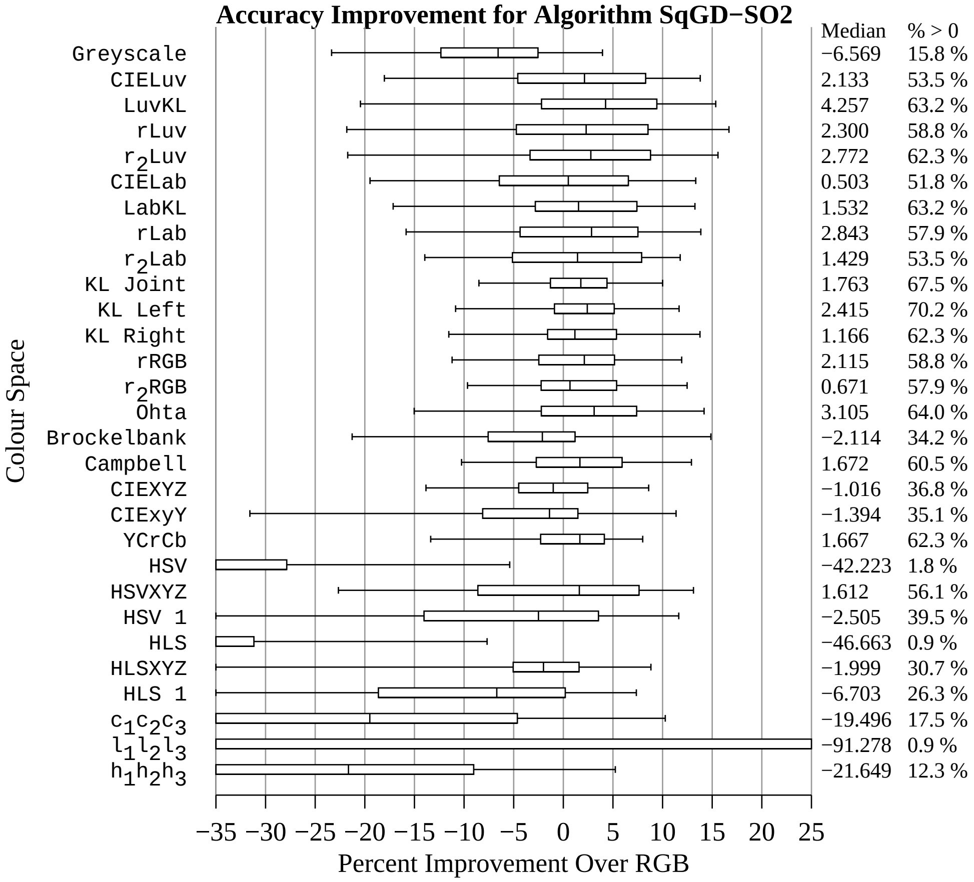


Figure K.23: Box plots of the over all aggregate percent accuracy improvement distributions for the SqGD-SO2 algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

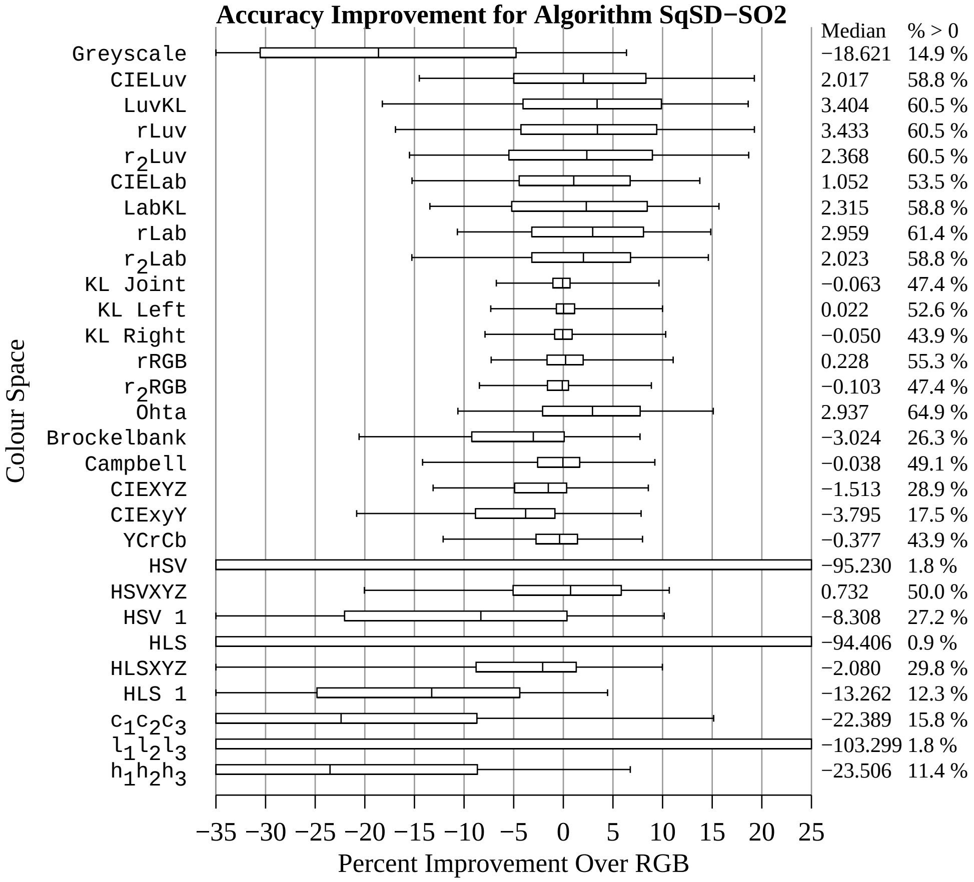


Figure K.24: Box plots of the over all aggregate percent accuracy improvement distributions for the SqSD-SO2 algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

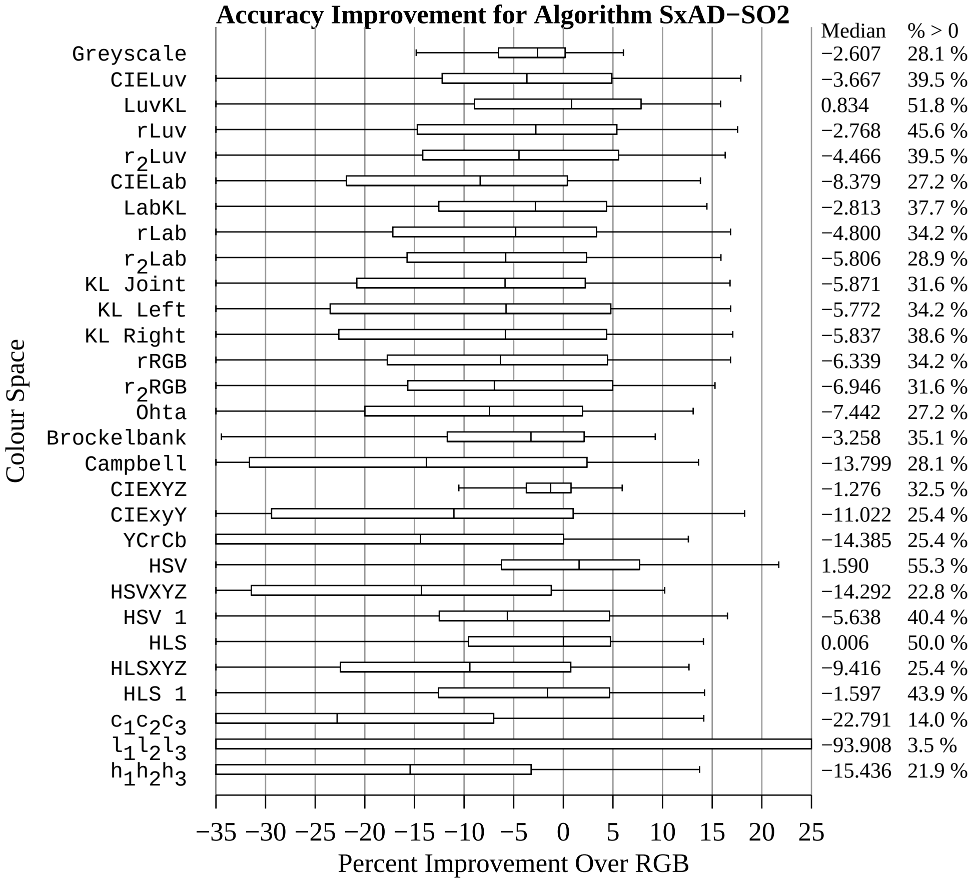


Figure K.25: Box plots of the over all aggregate percent accuracy improvement distributions for the SxAD-SO2 algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

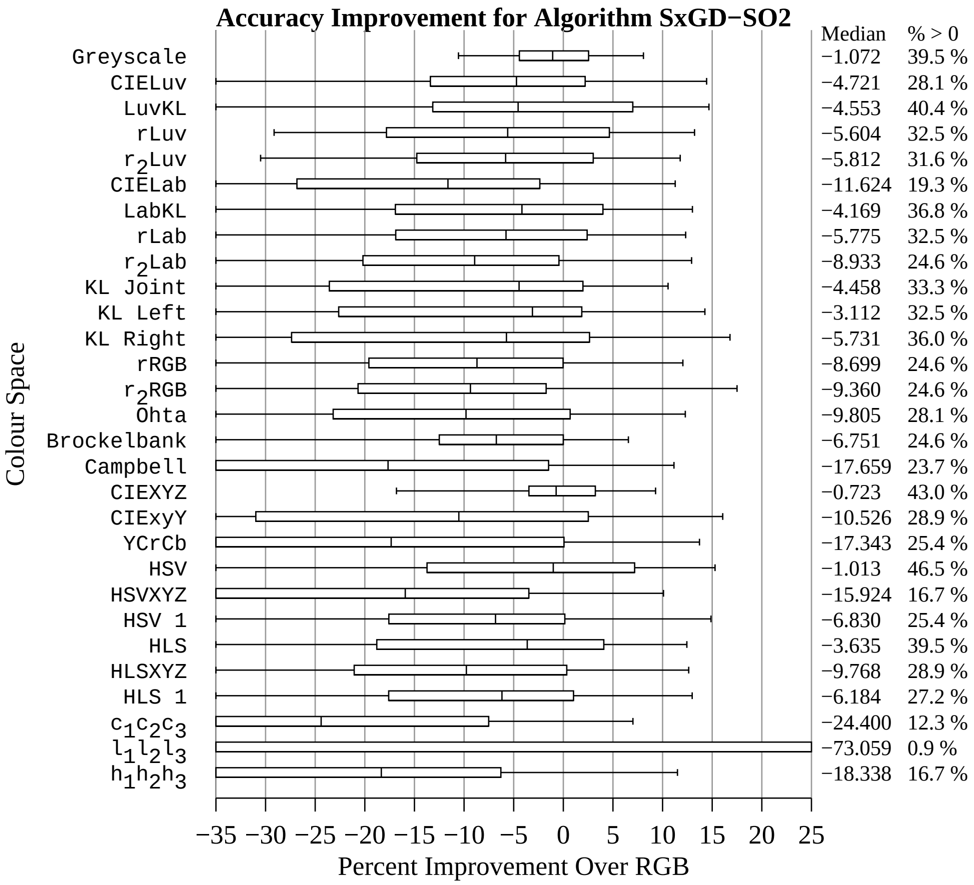


Figure K.26: Box plots of the over all aggregate percent accuracy improvement distributions for the SxGD-SO2 algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

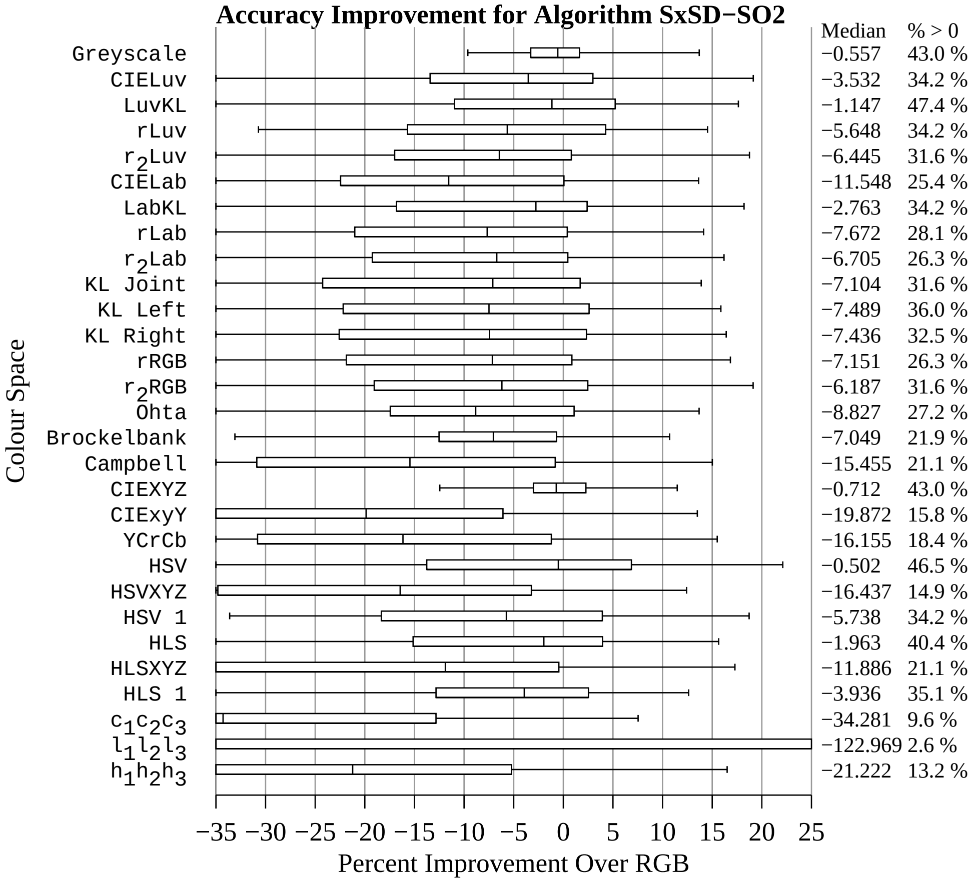


Figure K.27: Box plots of the over all aggregate percent accuracy improvement distributions for the SxSD-SO2 algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

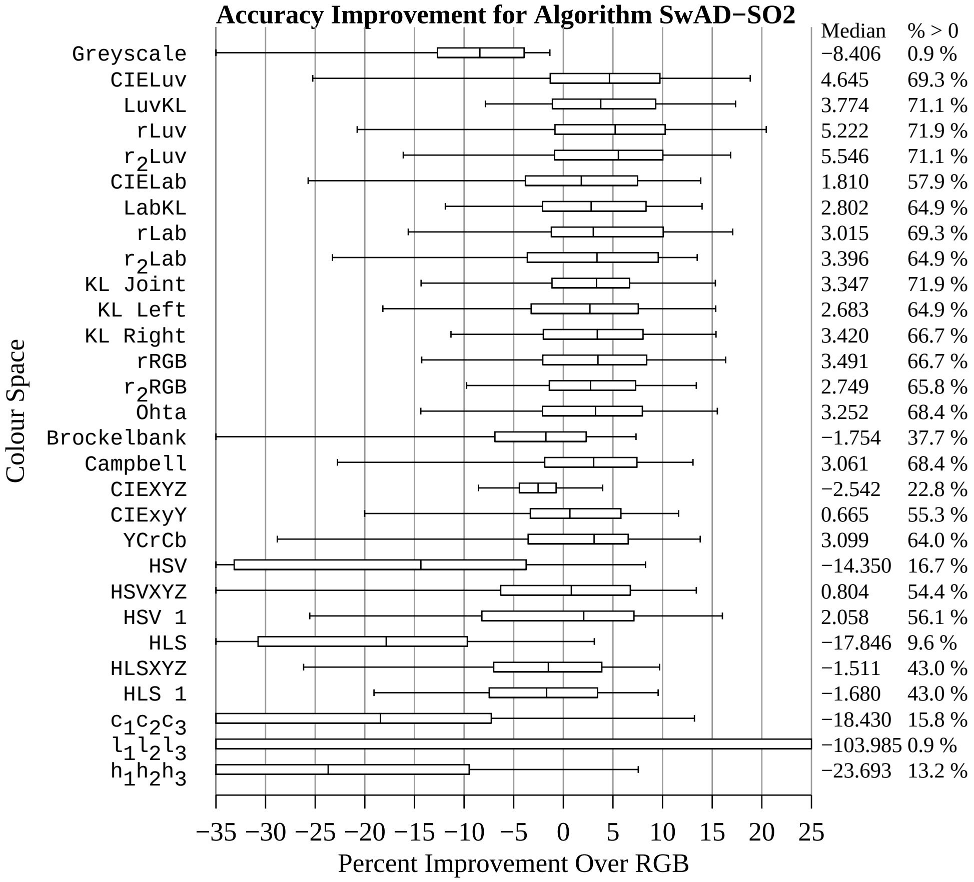


Figure K.28: Box plots of the over all aggregate percent accuracy improvement distributions for the SwAD-SO2 algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

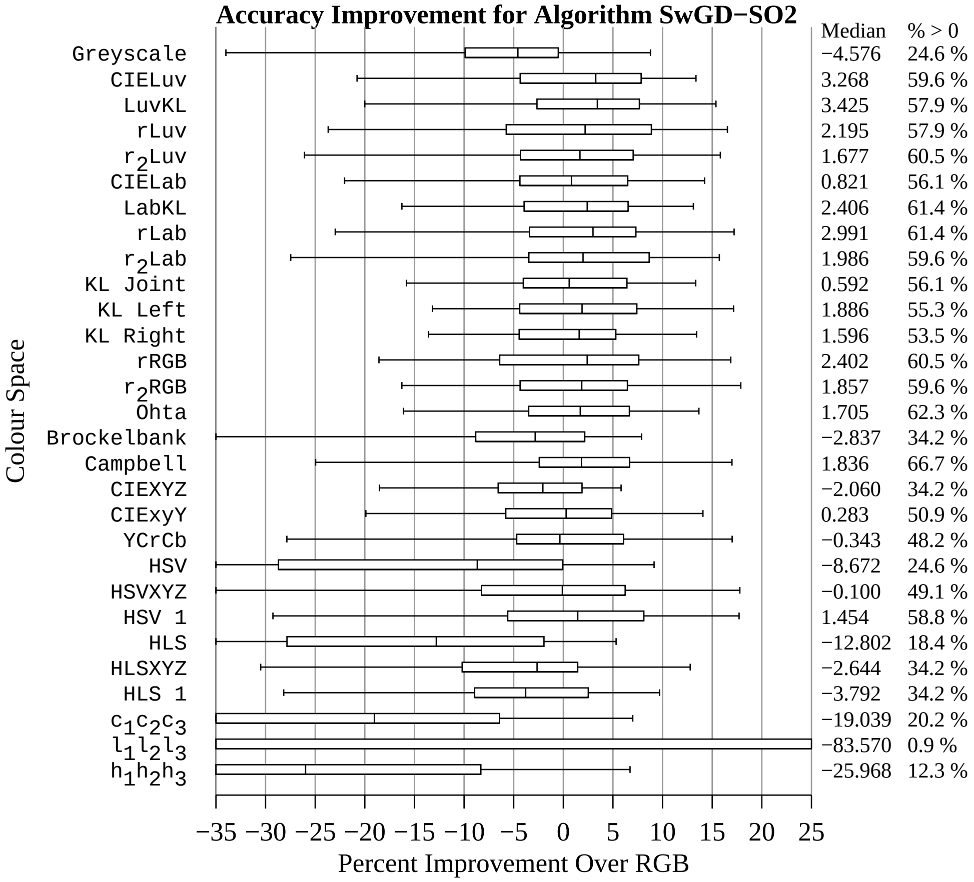


Figure K.29: Box plots of the over all aggregate percent accuracy improvement distributions for the SwGD-SO2 algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

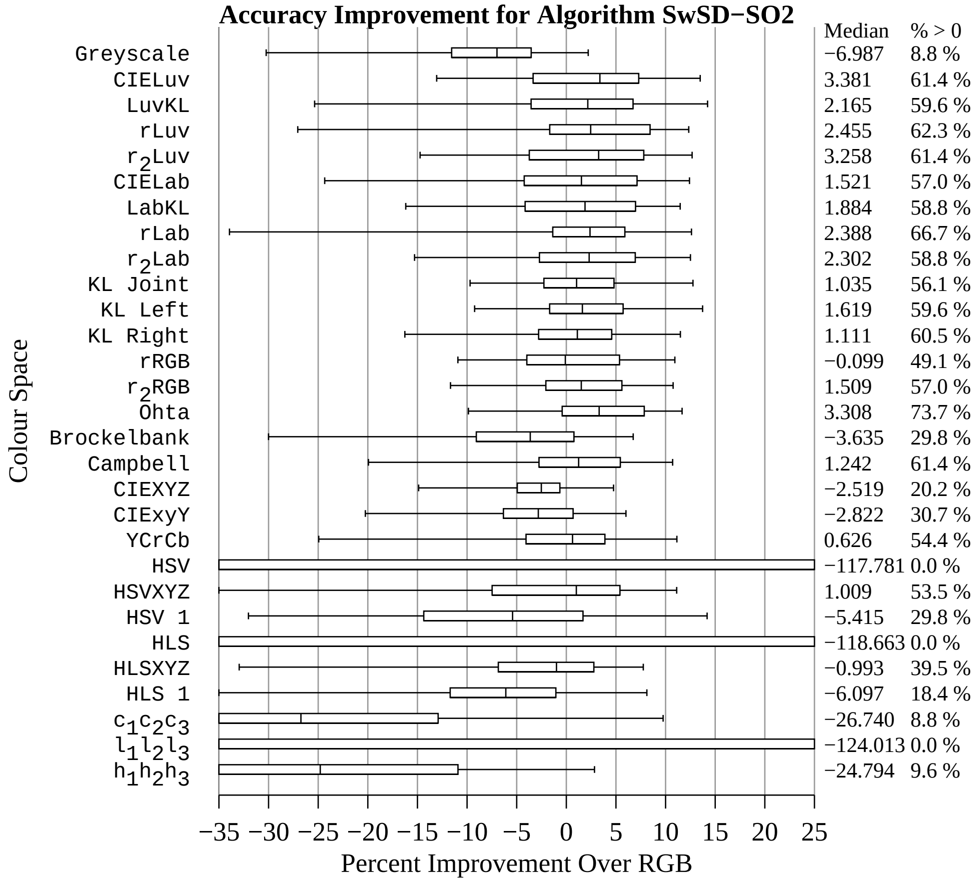


Figure K.30: Box plots of the over all aggregate percent accuracy improvement distributions for the SwSD-SO2 algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

K.3 RTree Framework Algorithms

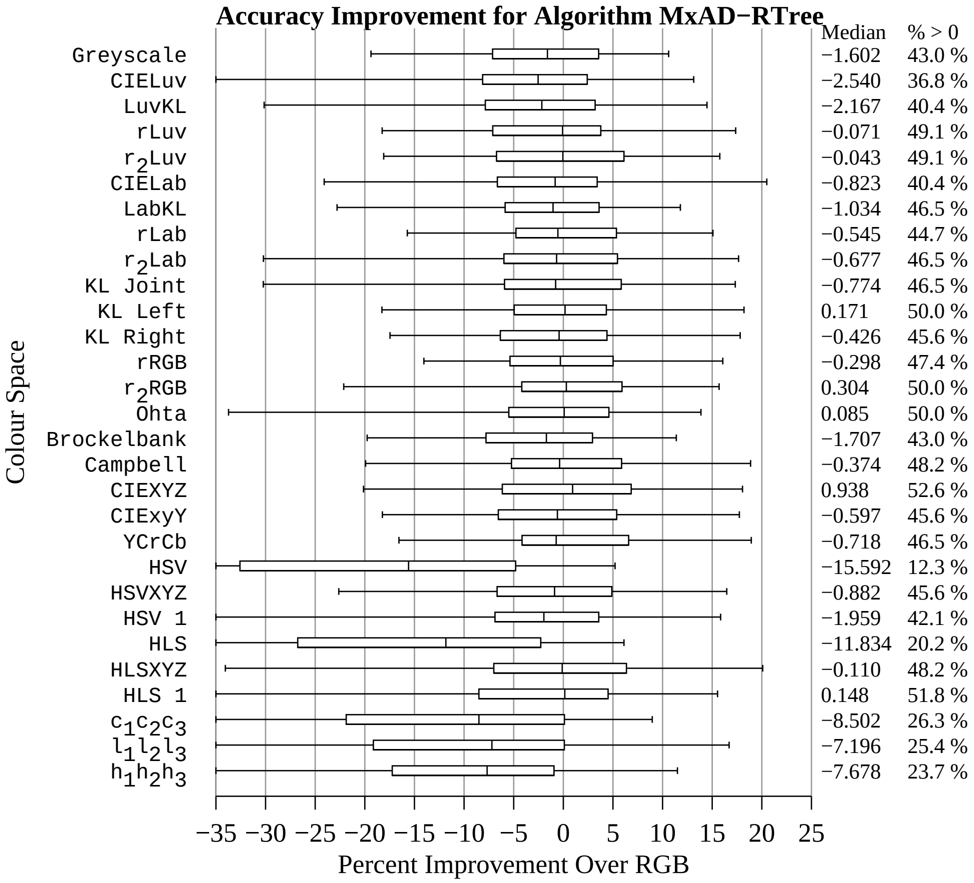


Figure K.31: Box plots of the over all aggregate percent accuracy improvement distributions for the MxAD-RTree algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

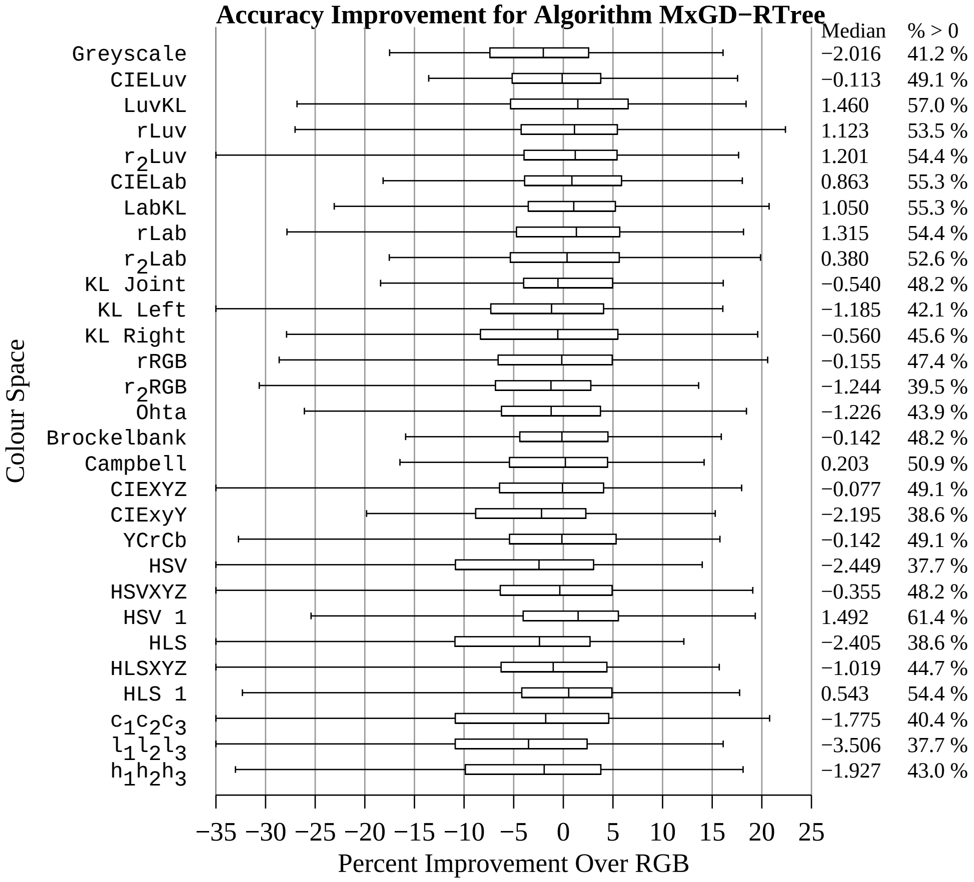


Figure K.32: Box plots of the over all aggregate percent accuracy improvement distributions for the MxGD-RTree algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

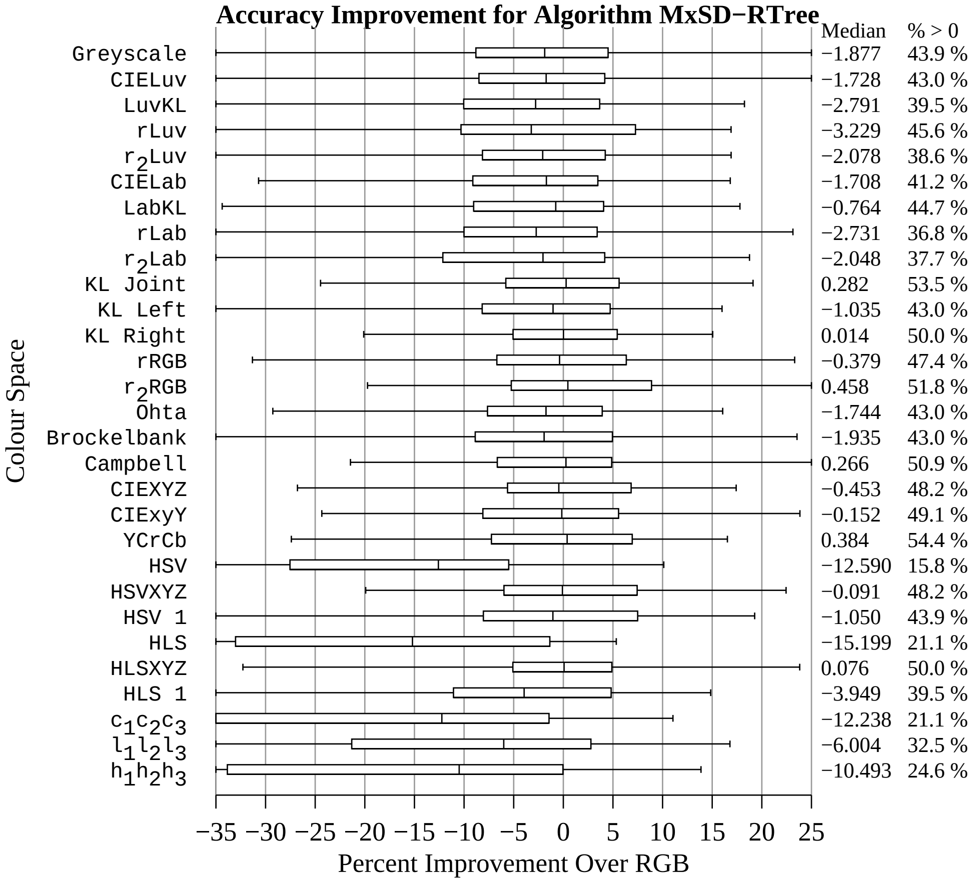


Figure K.33: Box plots of the over all aggregate percent accuracy improvement distributions for the MxSD-RTree algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

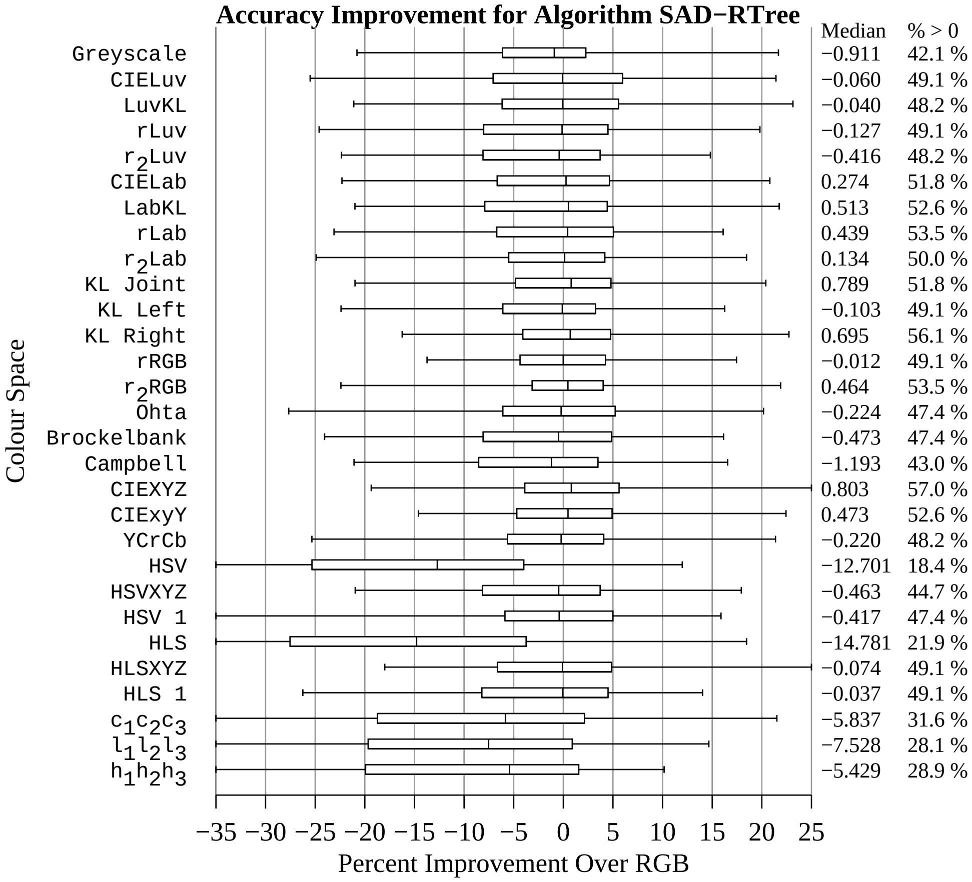


Figure K.34: Box plots of the over all aggregate percent accuracy improvement distributions for the SAD-RTree algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

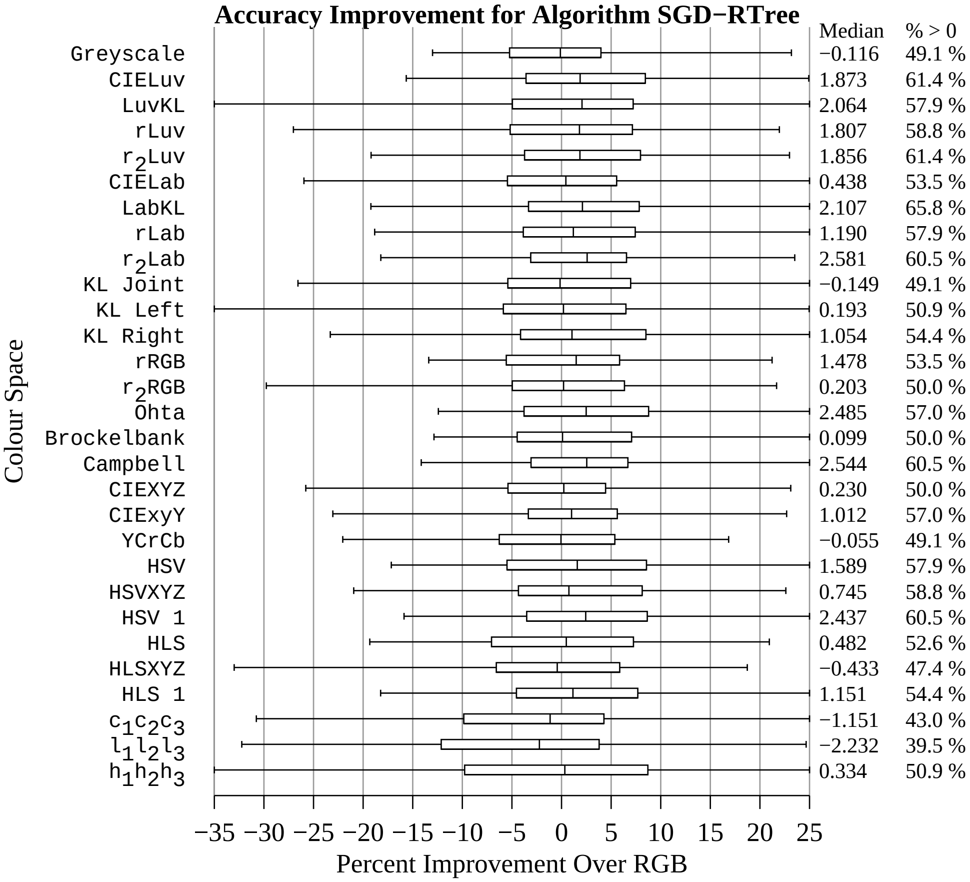


Figure K.35: Box plots of the over all aggregate percent accuracy improvement distributions for the SGD-RTree algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

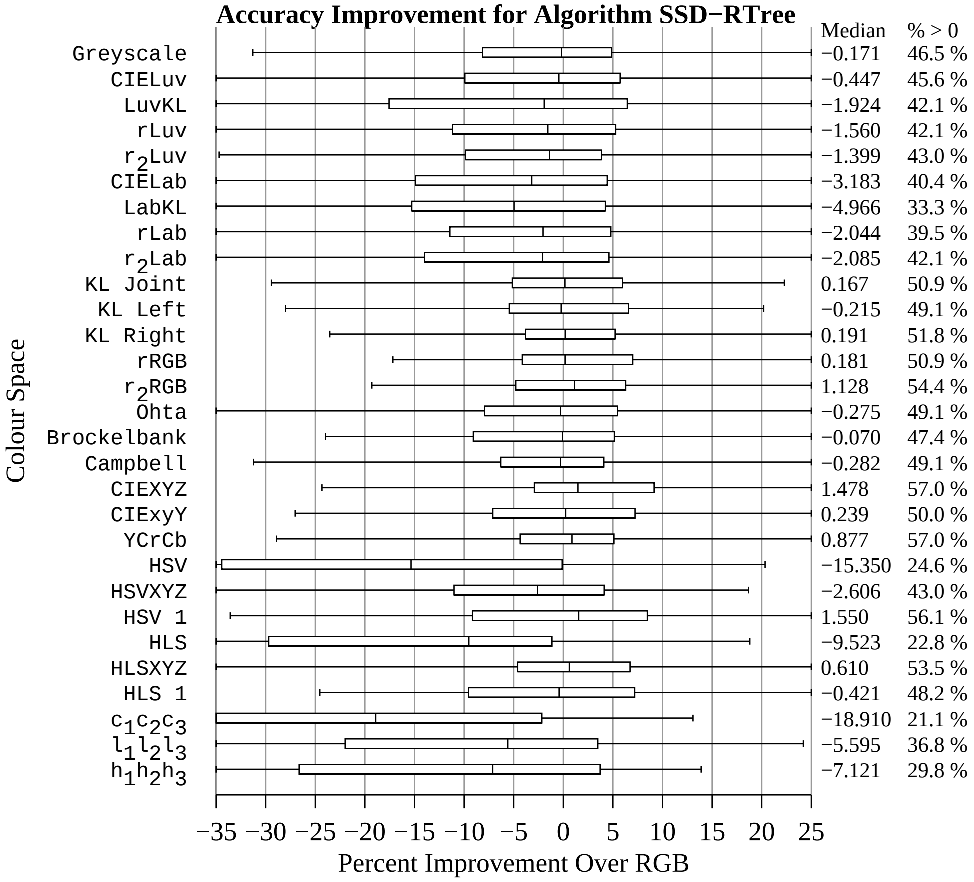


Figure K.36: Box plots of the over all aggregate percent accuracy improvement distributions for the SSD-RTree algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

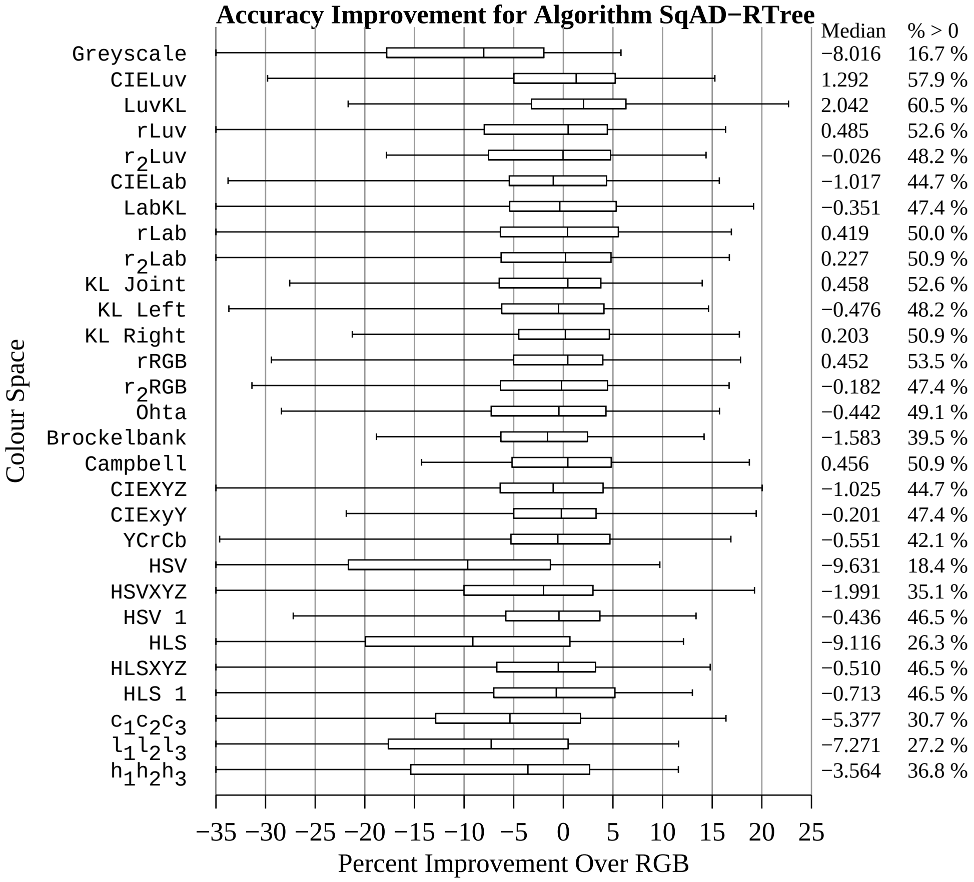


Figure K.37: Box plots of the over all aggregate percent accuracy improvement distributions for the SqAD-RTree algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

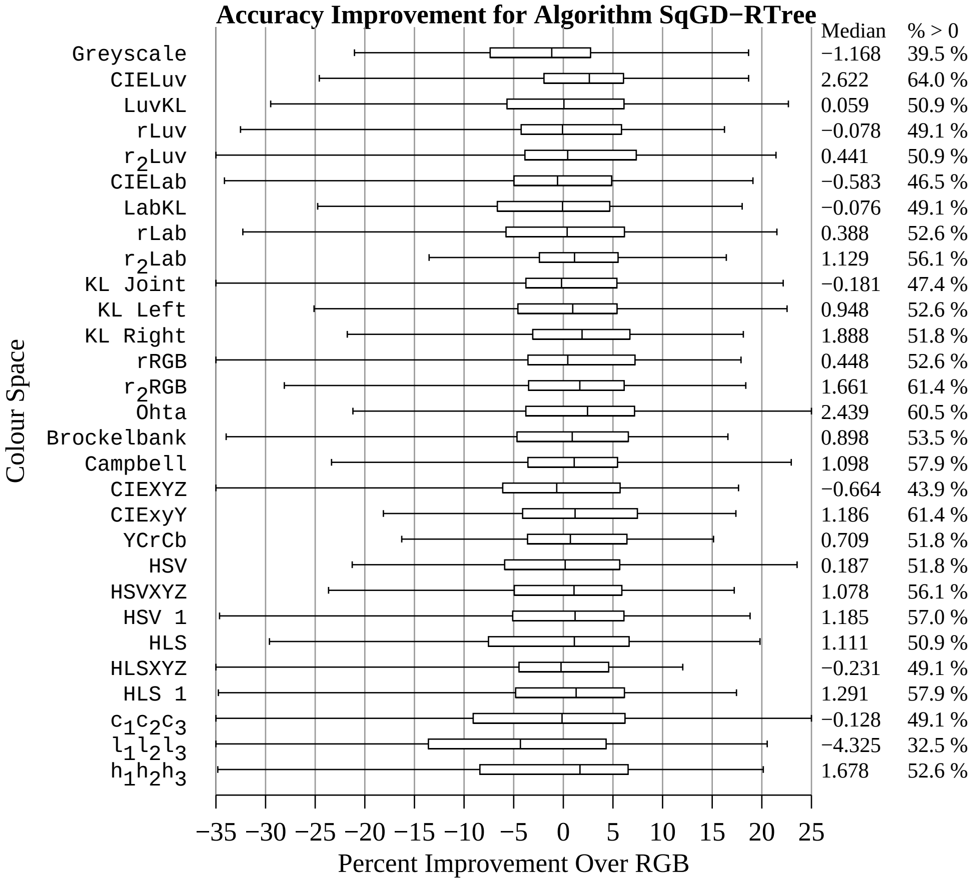
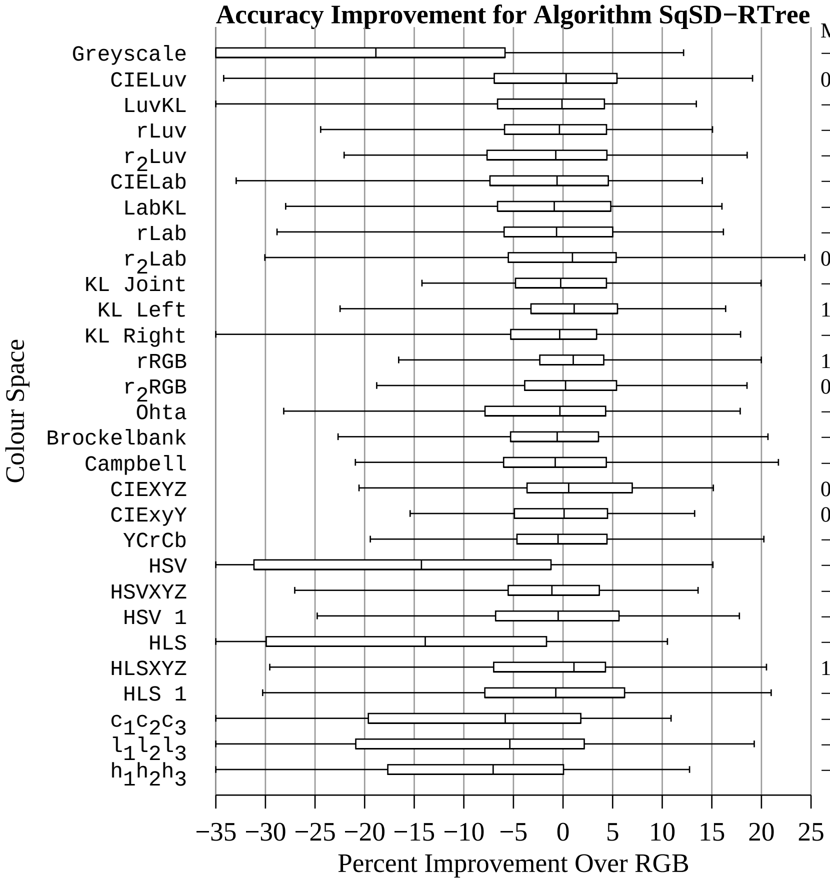


Figure K.38: Box plots of the over all aggregate percent accuracy improvement distributions for the SqGD-RTree algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.



	Median	% > 0
Greyscale	−18.865	12.3 %
CIELuv	0.311	51.8 %
LuvKL	−0.105	49.1 %
rLuv	−0.362	47.4 %
r_2Luv	−0.732	45.6 %
CIELab	−0.603	43.9 %
LabKL	−0.885	44.7 %
rLab	−0.653	45.6 %
r_2Lab	0.950	55.3 %
KL Joint	−0.236	48.2 %
KL Left	1.125	57.0 %
KL Right	−0.339	46.5 %
rRGB	1.030	60.5 %
r_2RGB	0.255	53.5 %
Ohta	−0.315	49.1 %
Brockelbank	−0.589	44.7 %
Campbell	−0.787	43.9 %
CIEXYZ	0.571	51.8 %
CIExyY	0.105	50.0 %
YCrCb	−0.504	48.2 %
HSV	−14.273	21.1 %
HSVXYZ	−1.120	43.0 %
HSV 1	−0.490	44.7 %
HLS	−13.890	21.1 %
HLSXYZ	1.103	53.5 %
HLS 1	−0.729	48.2 %
$c_1c_2c_3$	−5.824	29.8 %
$l_1l_2l_3$	−5.363	30.7 %
$h_1h_2h_3$	−7.046	26.3 %

Figure K.39: Box plots of the over all aggregate percent accuracy improvement distributions for the SqSD-RTree algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

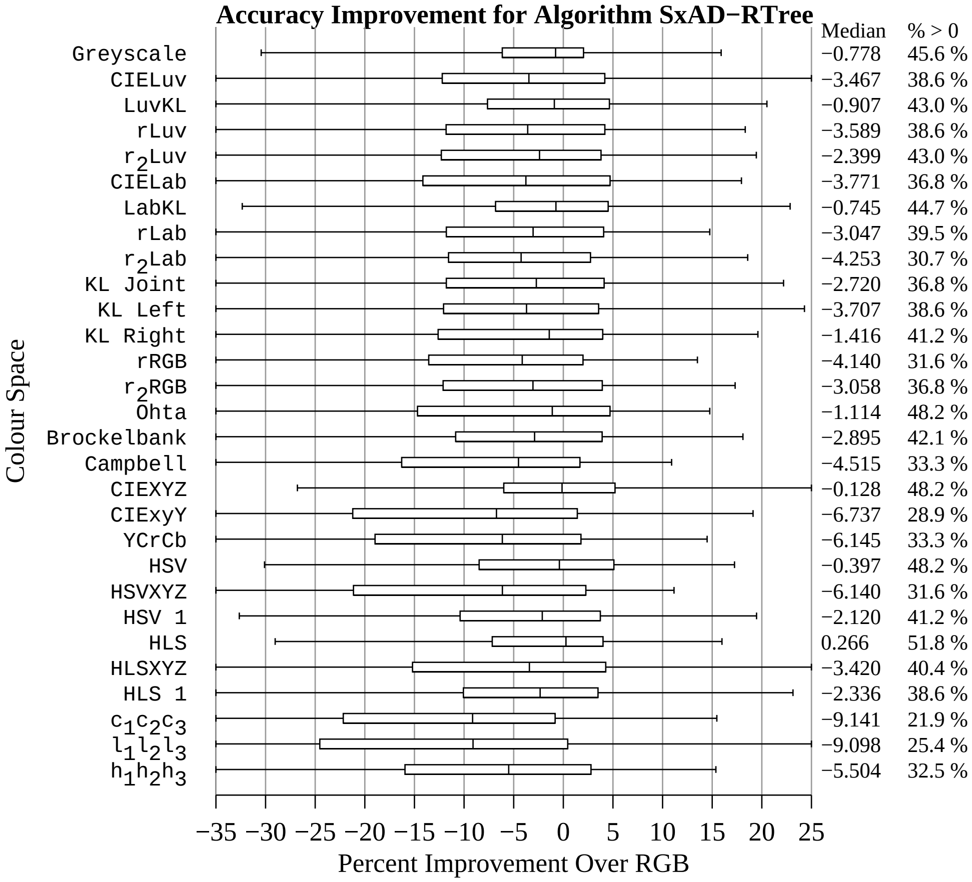


Figure K.40: Box plots of the over all aggregate percent accuracy improvement distributions for the SxAD-RTree algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

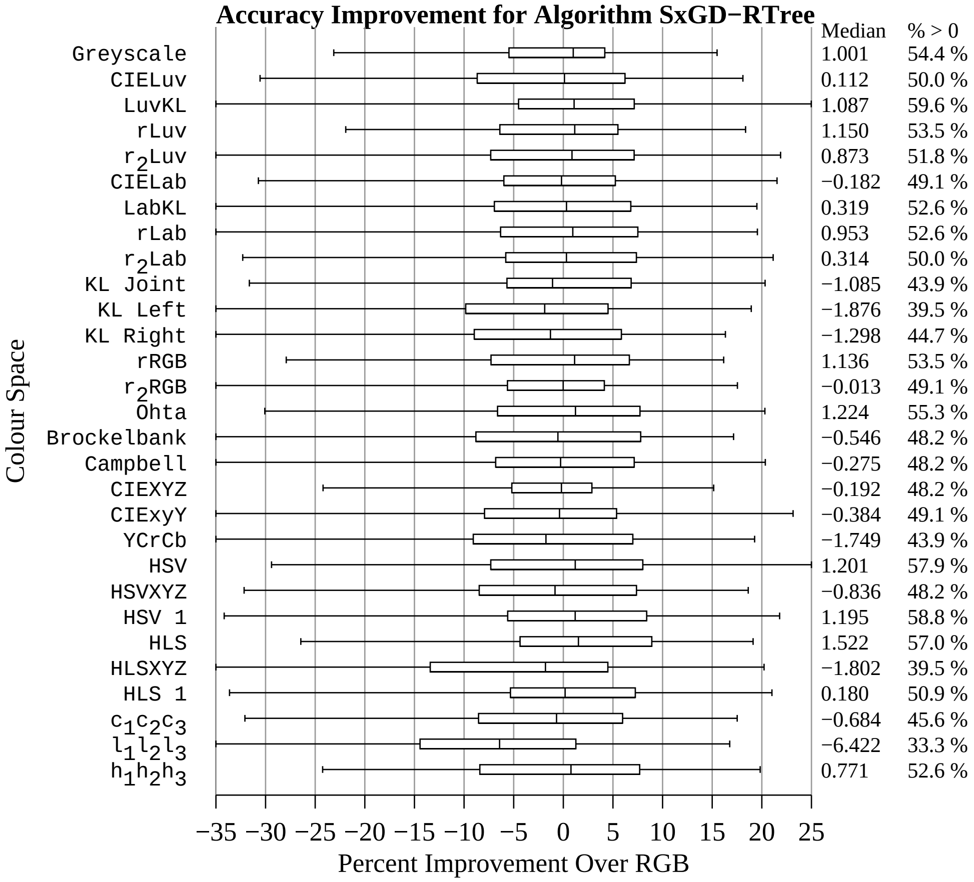


Figure K.41: Box plots of the over all aggregate percent accuracy improvement distributions for the SxGD-RTree algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

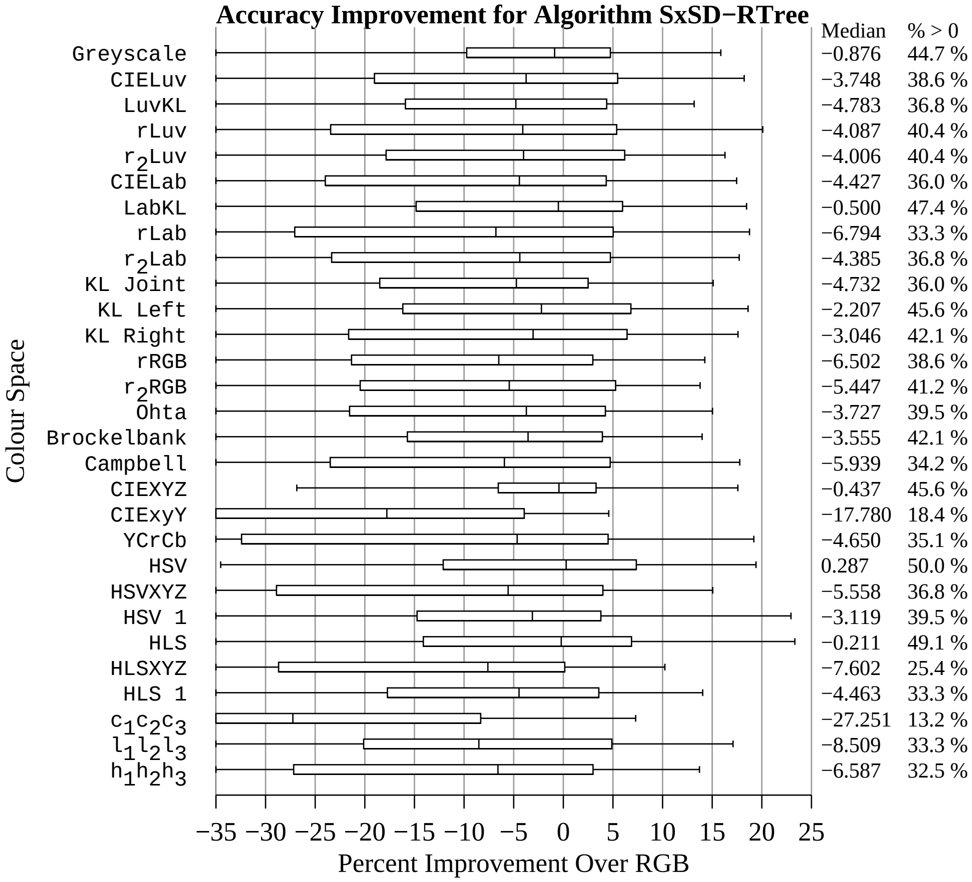


Figure K.42: Box plots of the over all aggregate percent accuracy improvement distributions for the SxSD-RTree algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

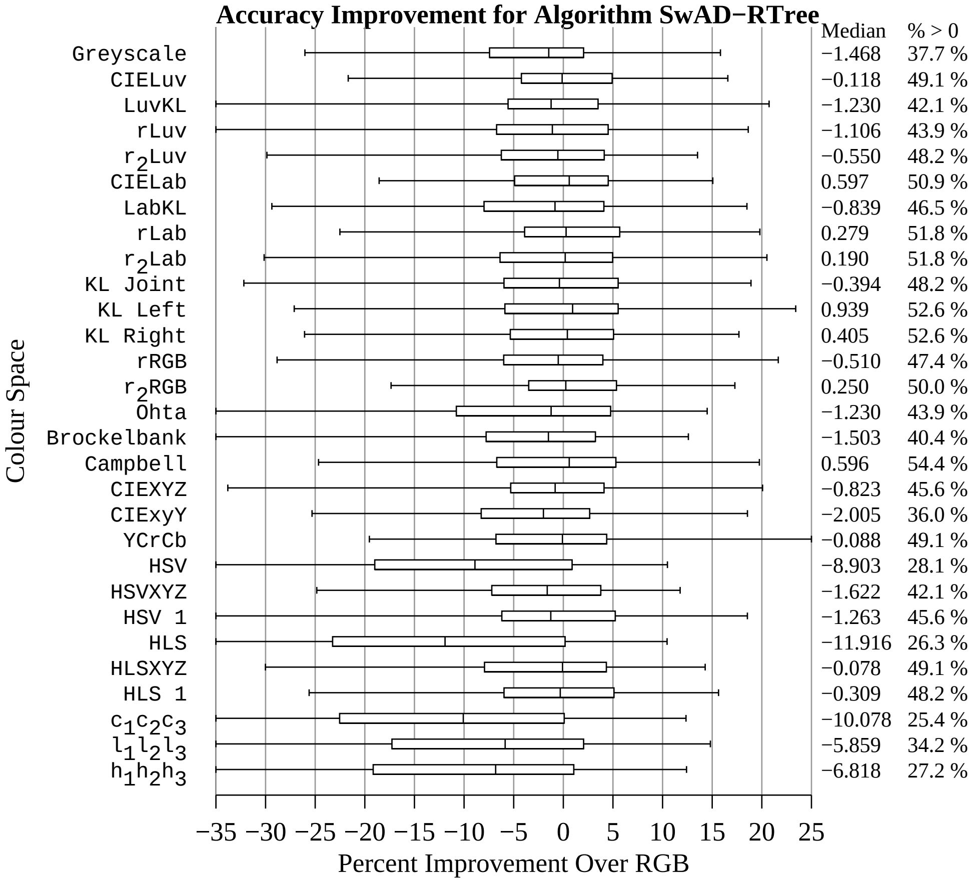


Figure K.43: Box plots of the over all aggregate percent accuracy improvement distributions for the SwAD-RTree algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

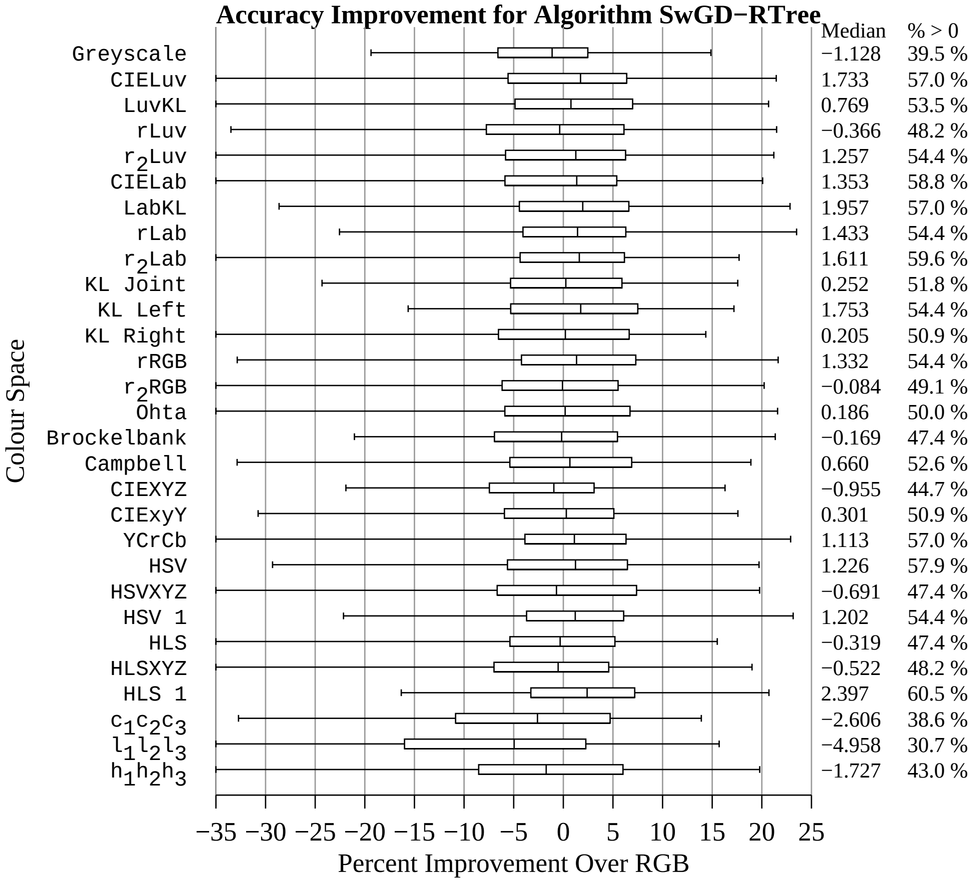


Figure K.44: Box plots of the over all aggregate percent accuracy improvement distributions for the SwGD-RTree algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

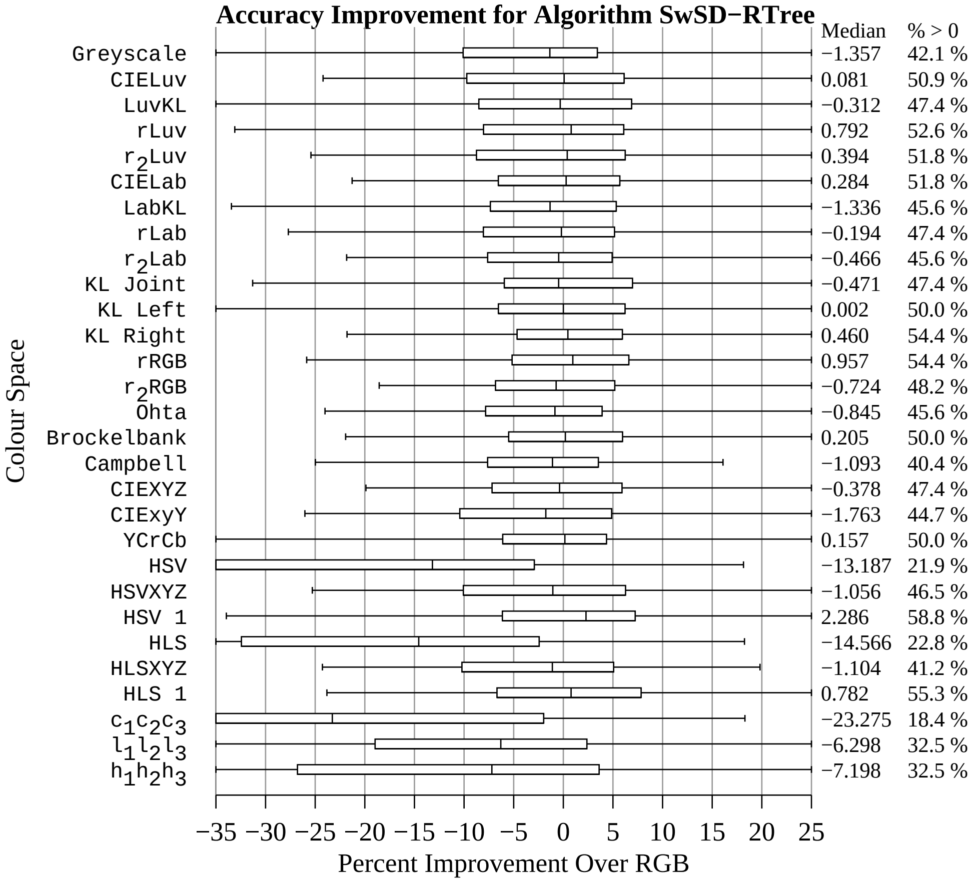


Figure K.45: Box plots of the over all aggregate percent accuracy improvement distributions for the SwSD-RTree algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

K.4 HBPTL Framework Algorithms

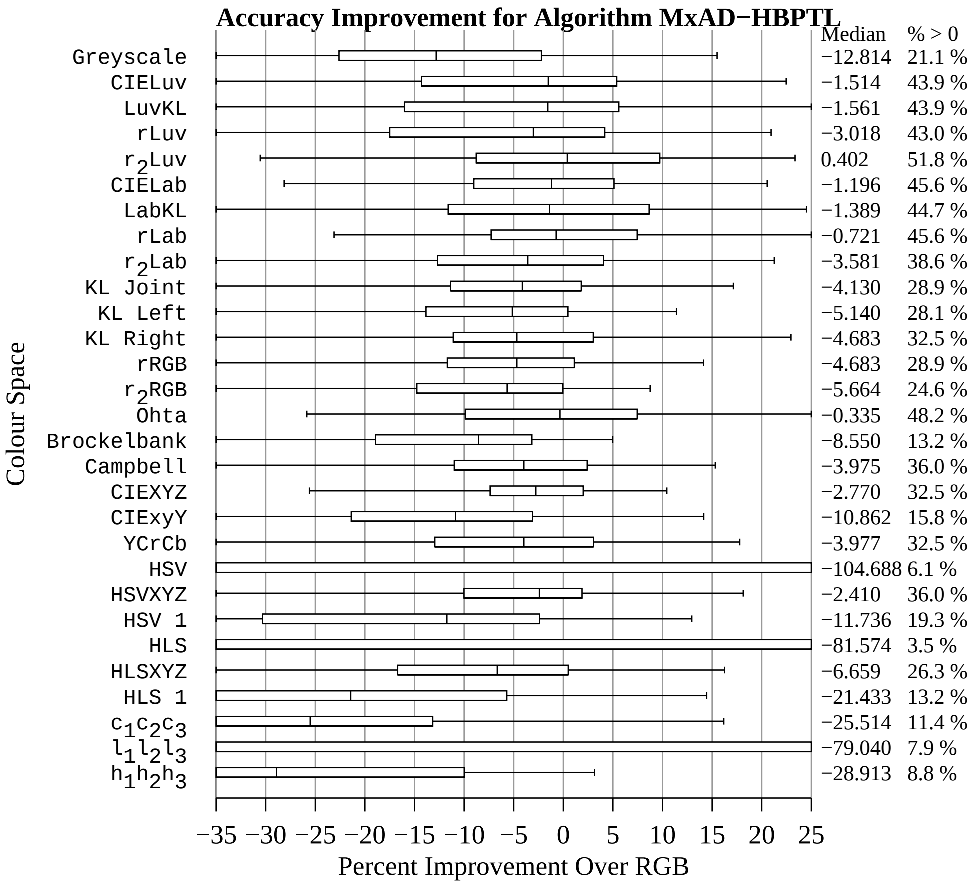


Figure K.46: Box plots of the over all aggregate percent accuracy improvement distributions for the MxAD-HBPTL algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

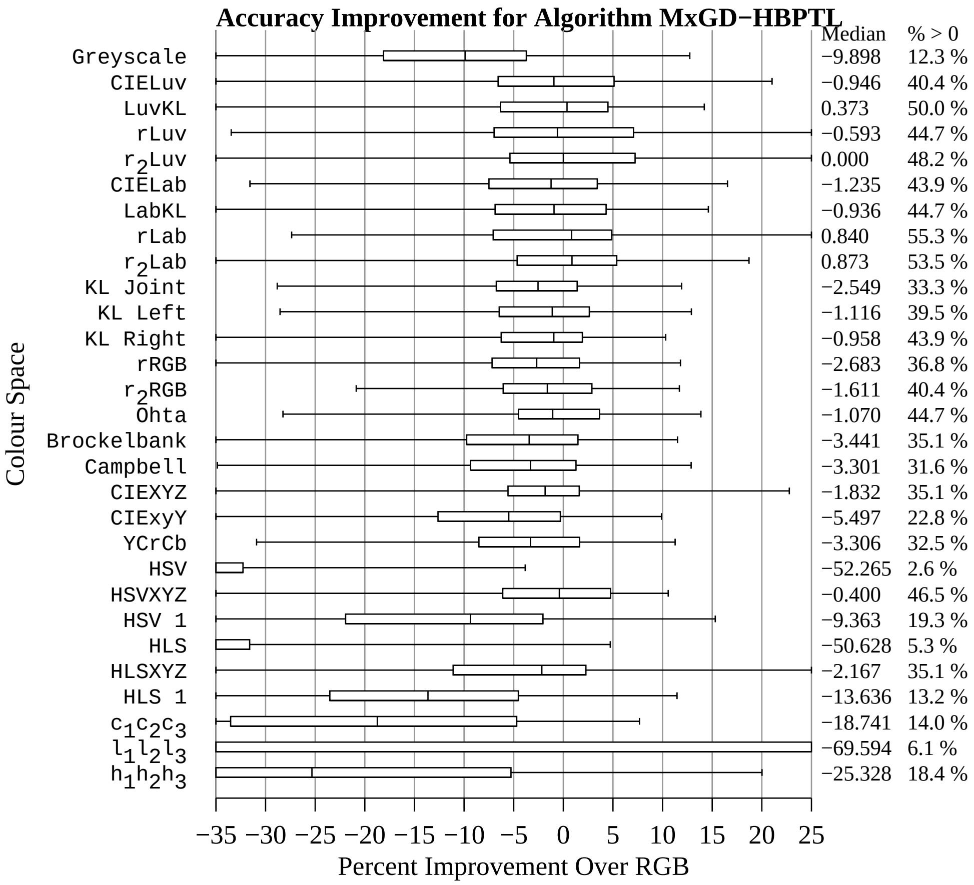


Figure K.47: Box plots of the over all aggregate percent accuracy improvement distributions for the MxGD-HBPTL algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

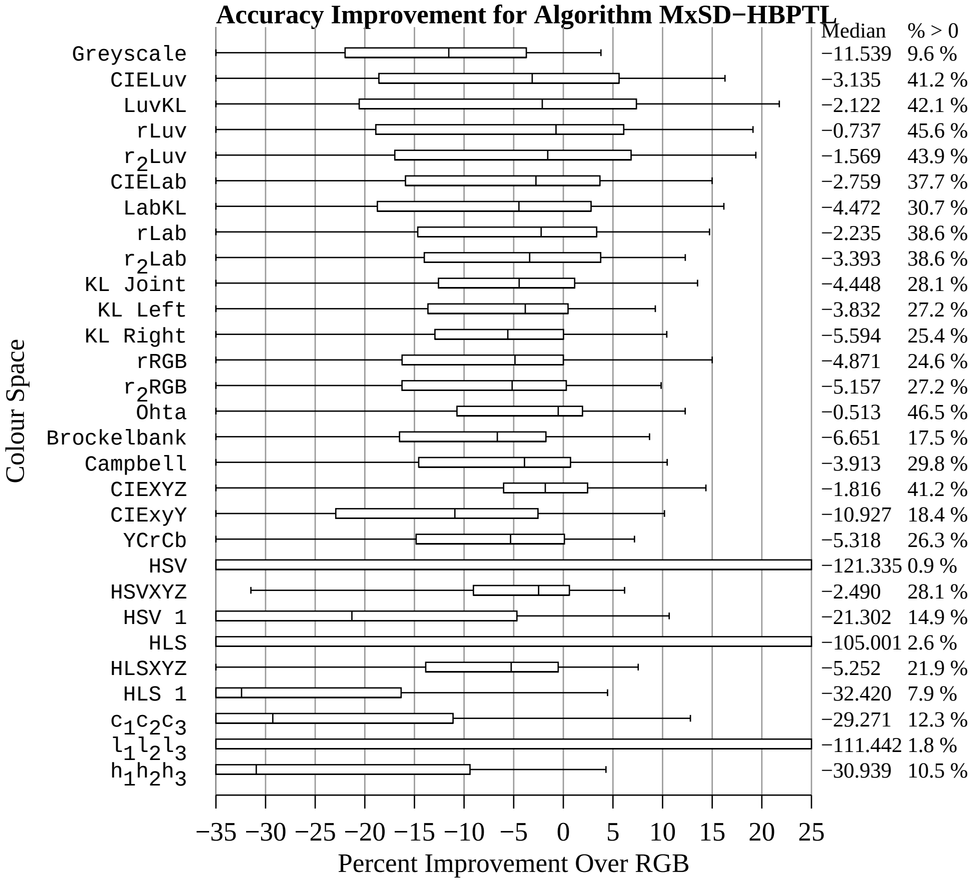


Figure K.48: Box plots of the over all aggregate percent accuracy improvement distributions for the MxSD-HBPTL algorithm for each colour representation. The plots indicate the fifth, 25th, 50th, 75th, and 95th percentiles of the distribution.

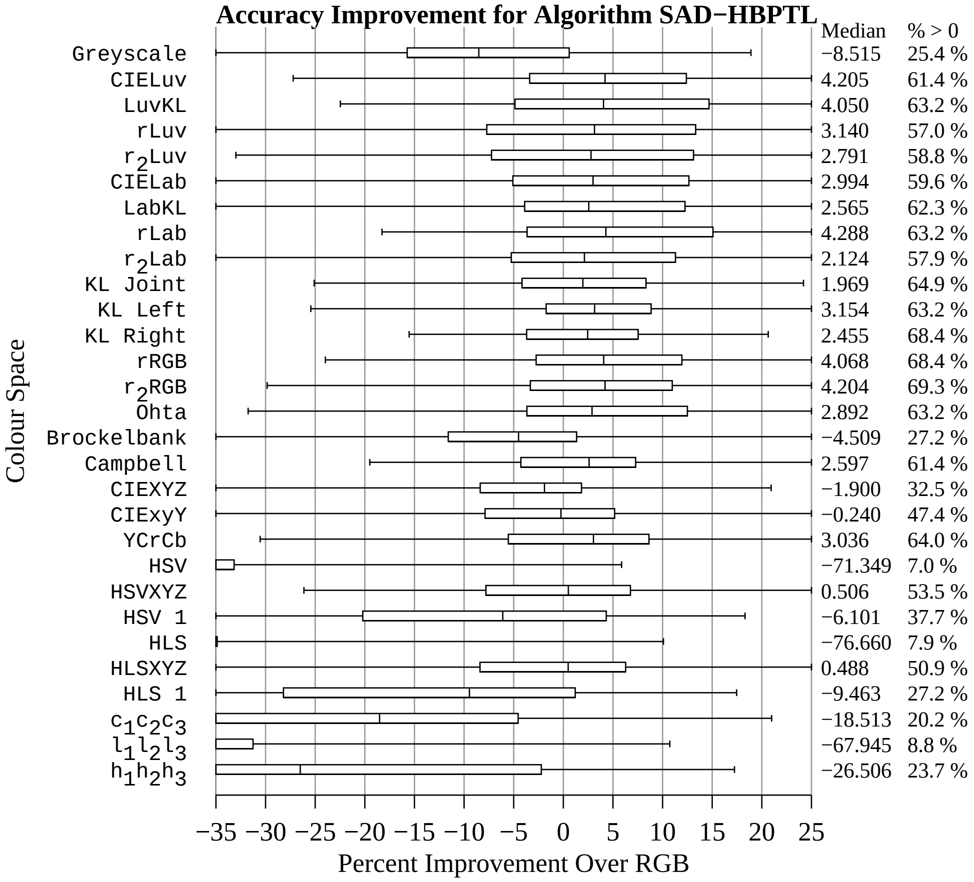


Figure K.49: Box plots of the over all aggregate percent accuracy improvement distributions for the SAD-HBPTL algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

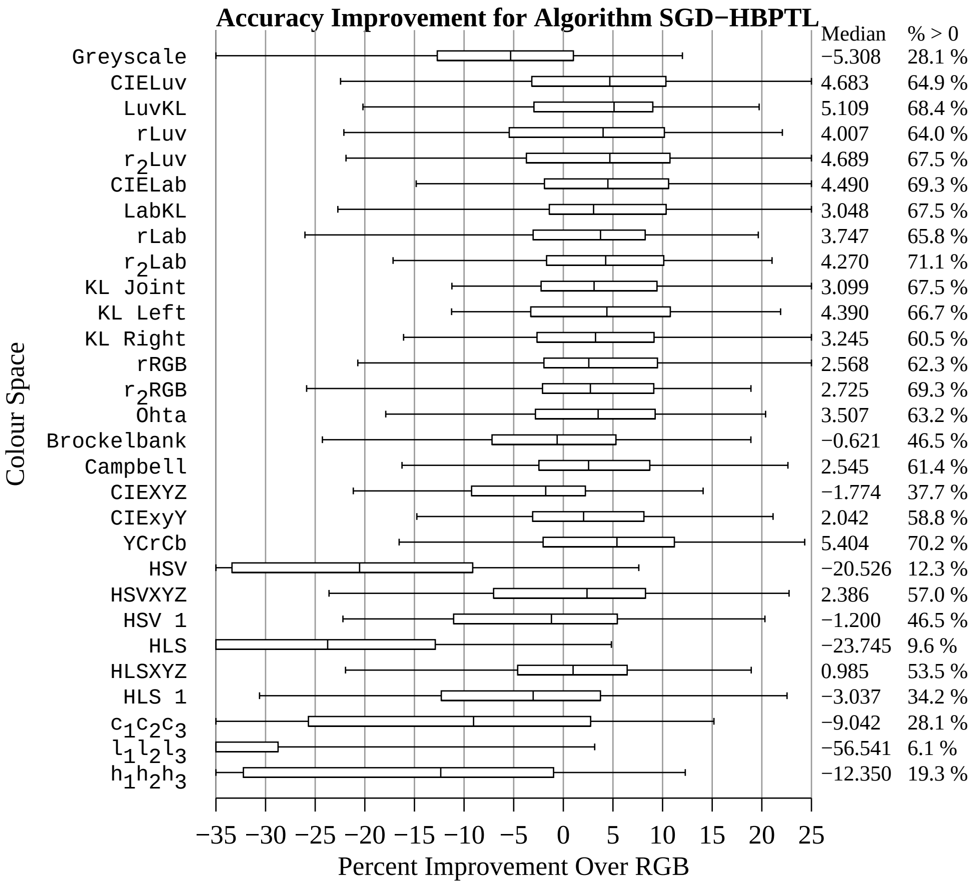


Figure K.50: Box plots of the over all aggregate percent accuracy improvement distributions for the SGD-HBPTL algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

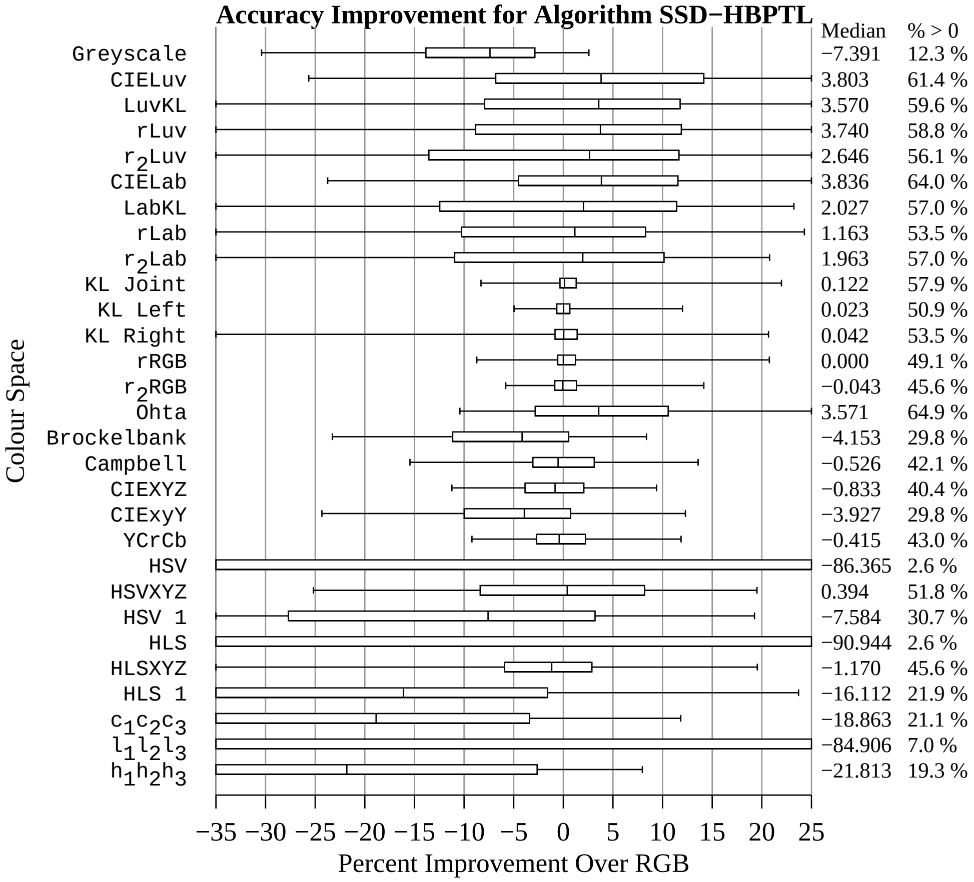


Figure K.51: Box plots of the over all aggregate percent accuracy improvement distributions for the SSD-HBPTL algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

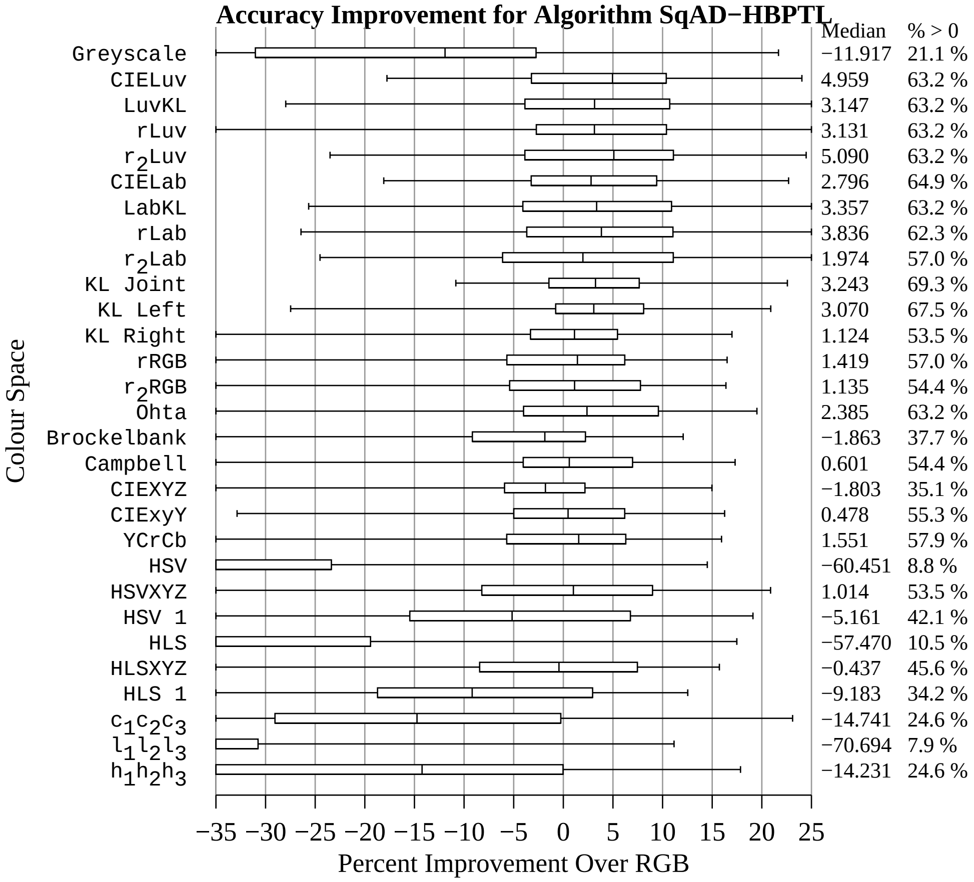


Figure K.52: Box plots of the over all aggregate percent accuracy improvement distributions for the SqAD-HBPTL algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

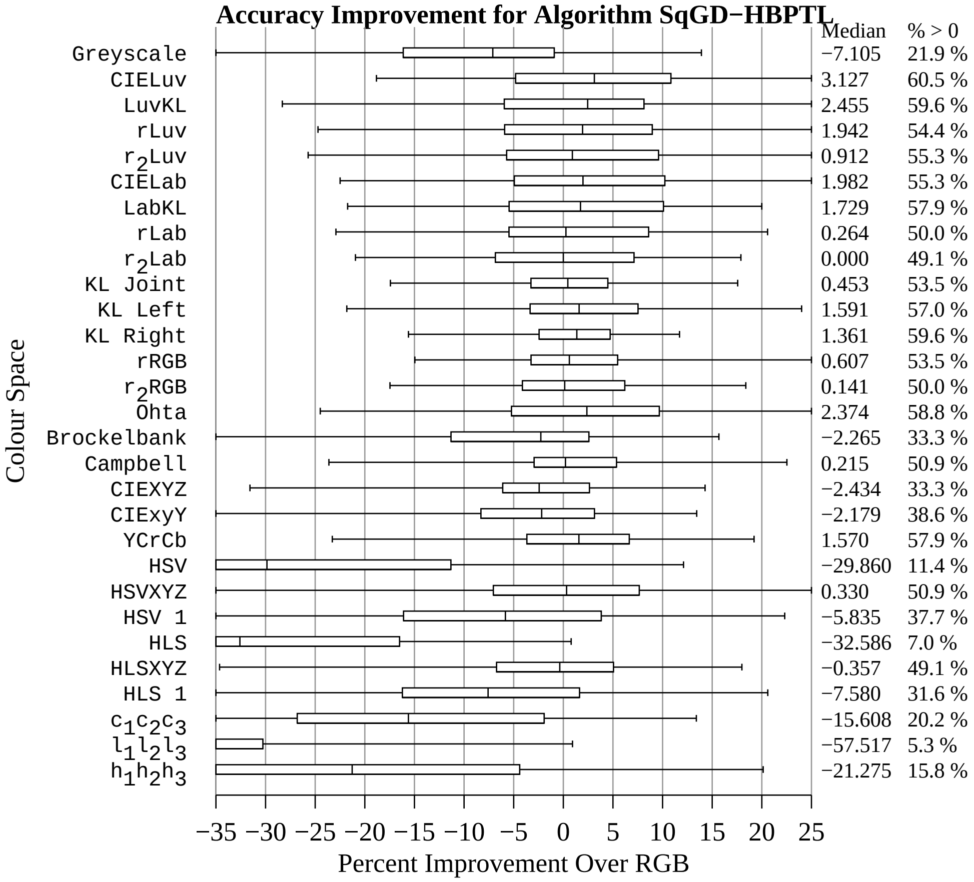


Figure K.53: Box plots of the over all aggregate percent accuracy improvement distributions for the SqGD-HBPTL algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

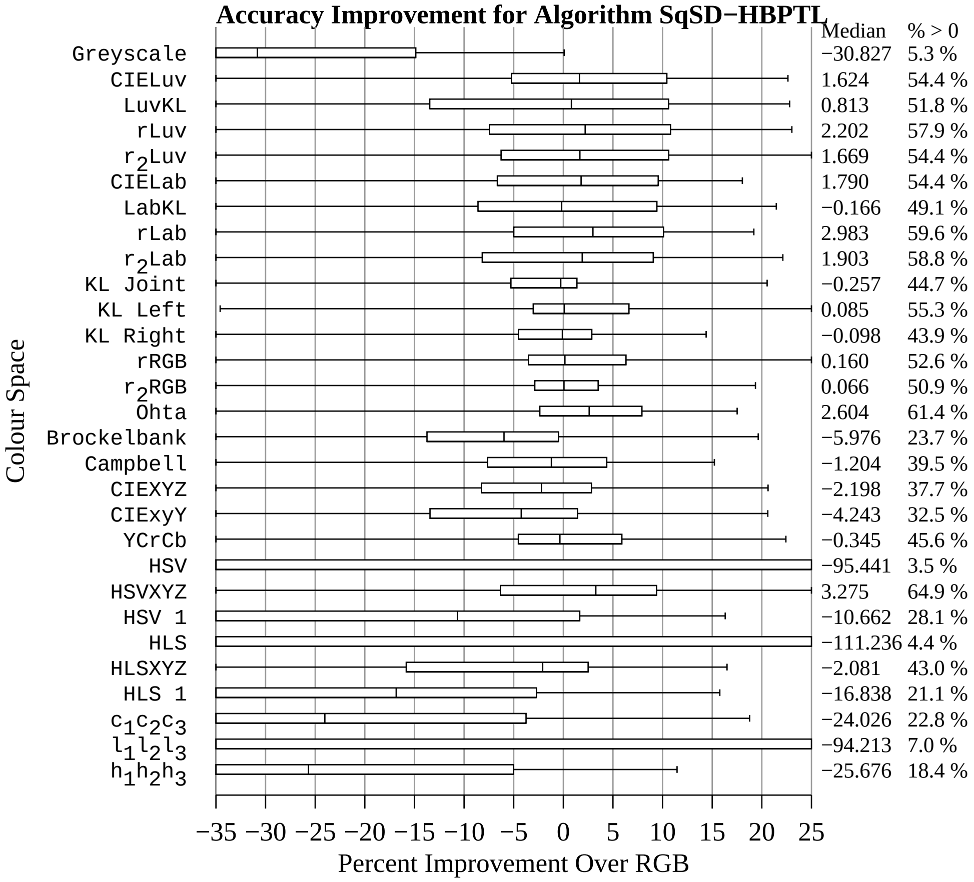


Figure K.54: Box plots of the over all aggregate percent accuracy improvement distributions for the SqSD-HBPTL algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

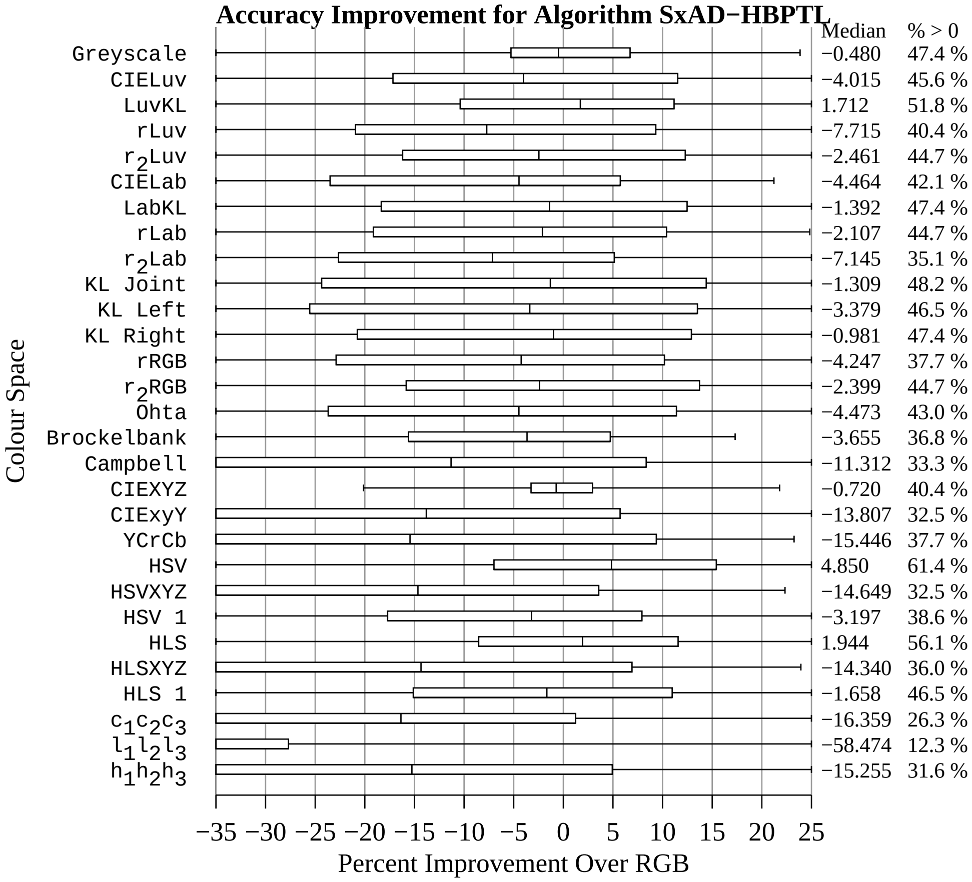


Figure K.55: Box plots of the over all aggregate percent accuracy improvement distributions for the SxAD-HBPTL algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

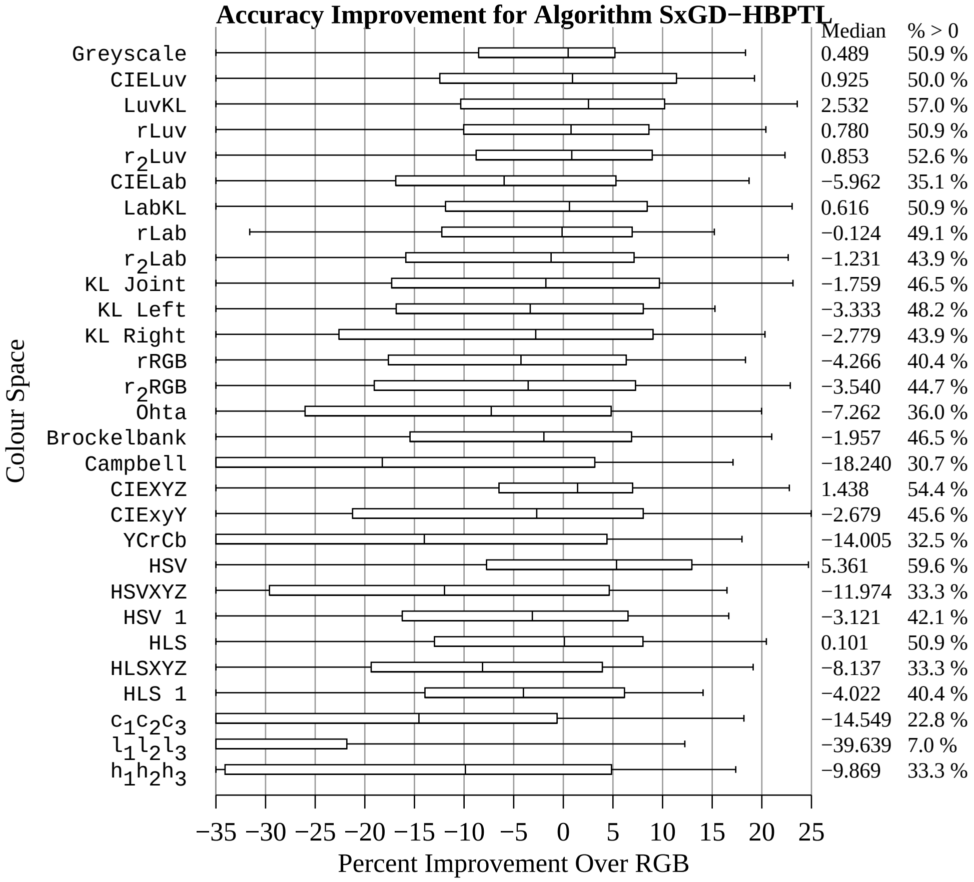


Figure K.56: Box plots of the over all aggregate percent accuracy improvement distributions for the SxGD-HBPTL algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

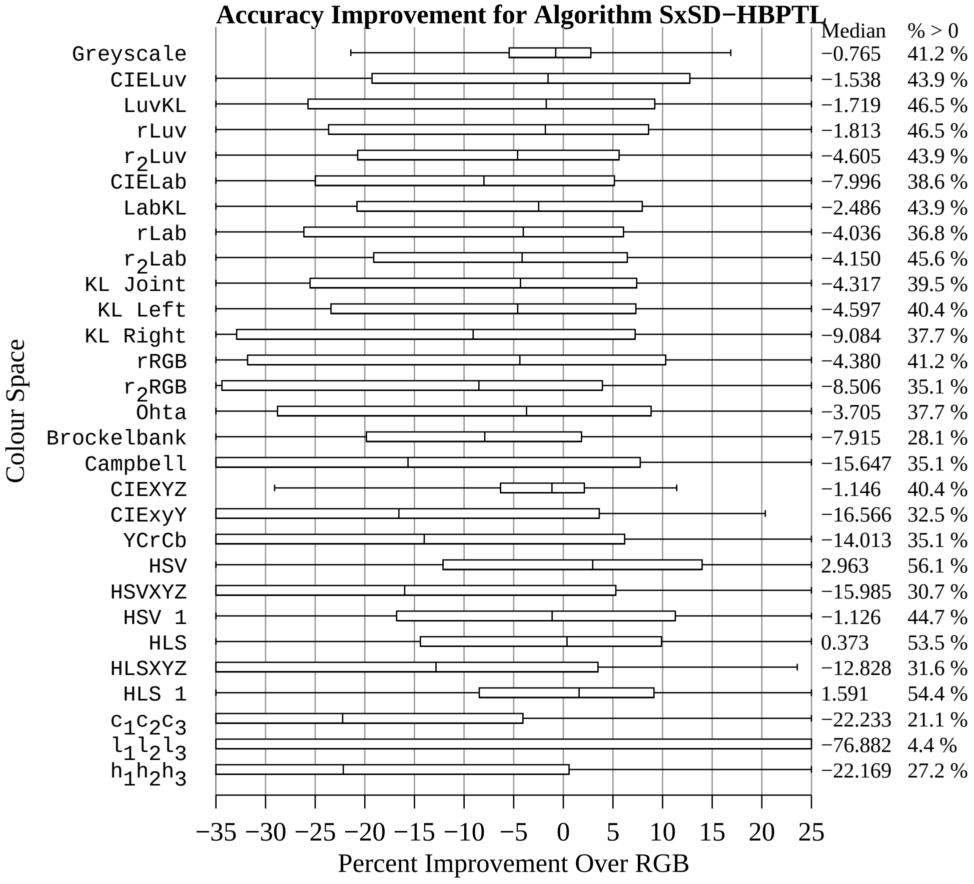


Figure K.57: Box plots of the over all aggregate percent accuracy improvement distributions for the SxSD-HBPTL algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

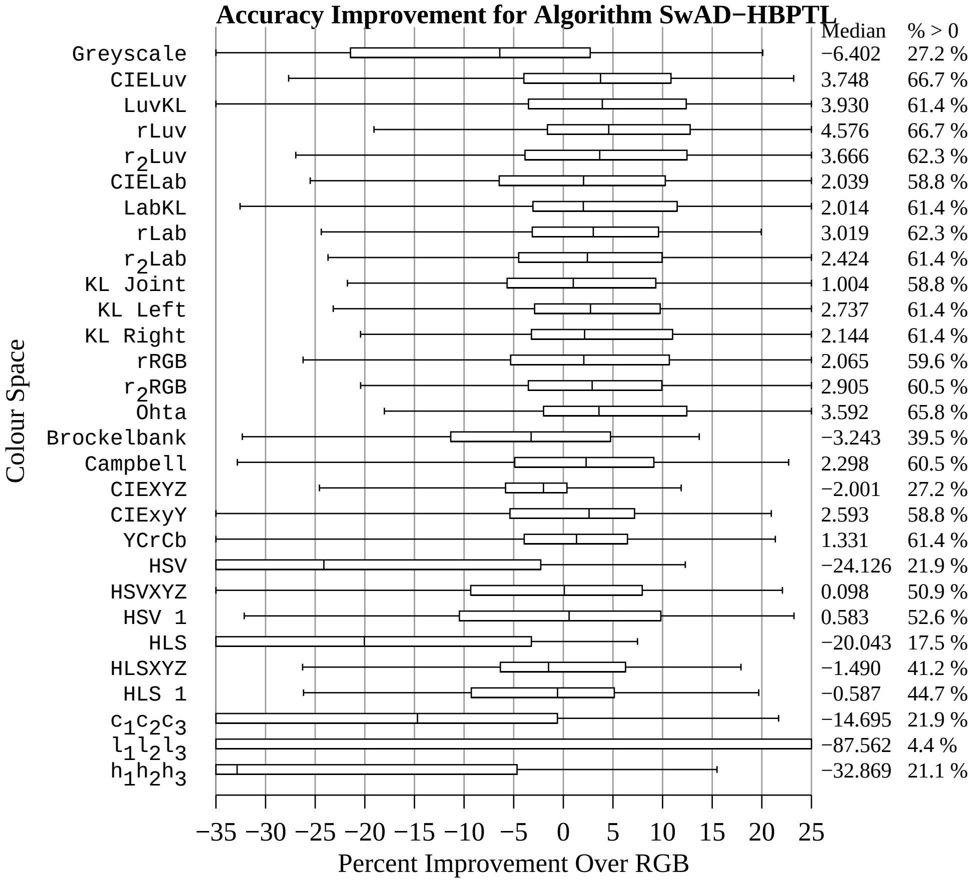


Figure K.58: Box plots of the over all aggregate percent accuracy improvement distributions for the SwAD-HBPTL algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

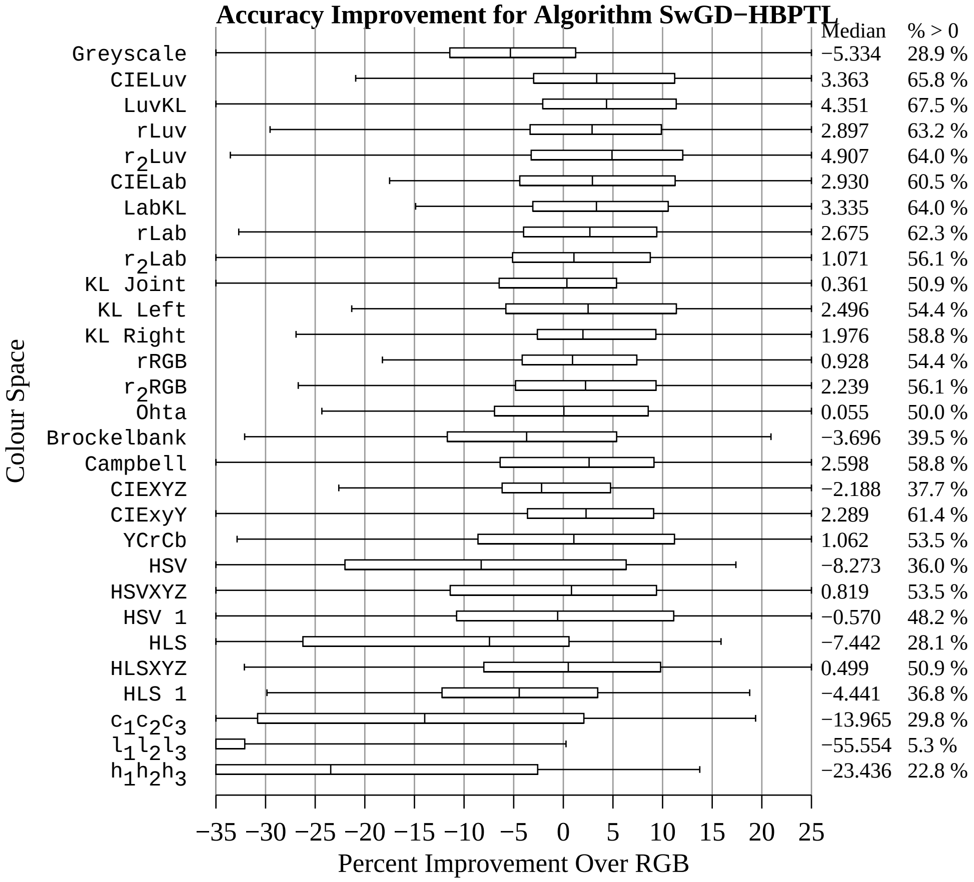


Figure K.59: Box plots of the over all aggregate percent accuracy improvement distributions for the SwGD-HBPTL algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

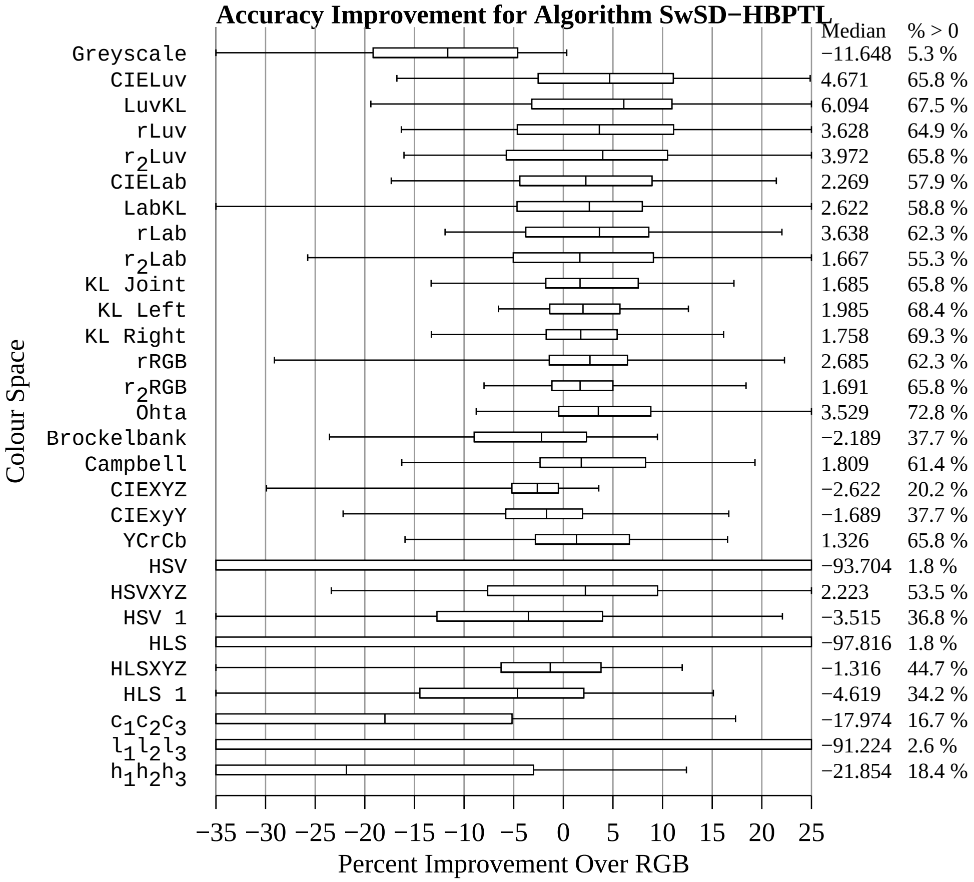


Figure K.60: Box plots of the over all aggregate percent accuracy improvement distributions for the SwSD-HBPTL algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

K.5 HBPG Framework Algorithms

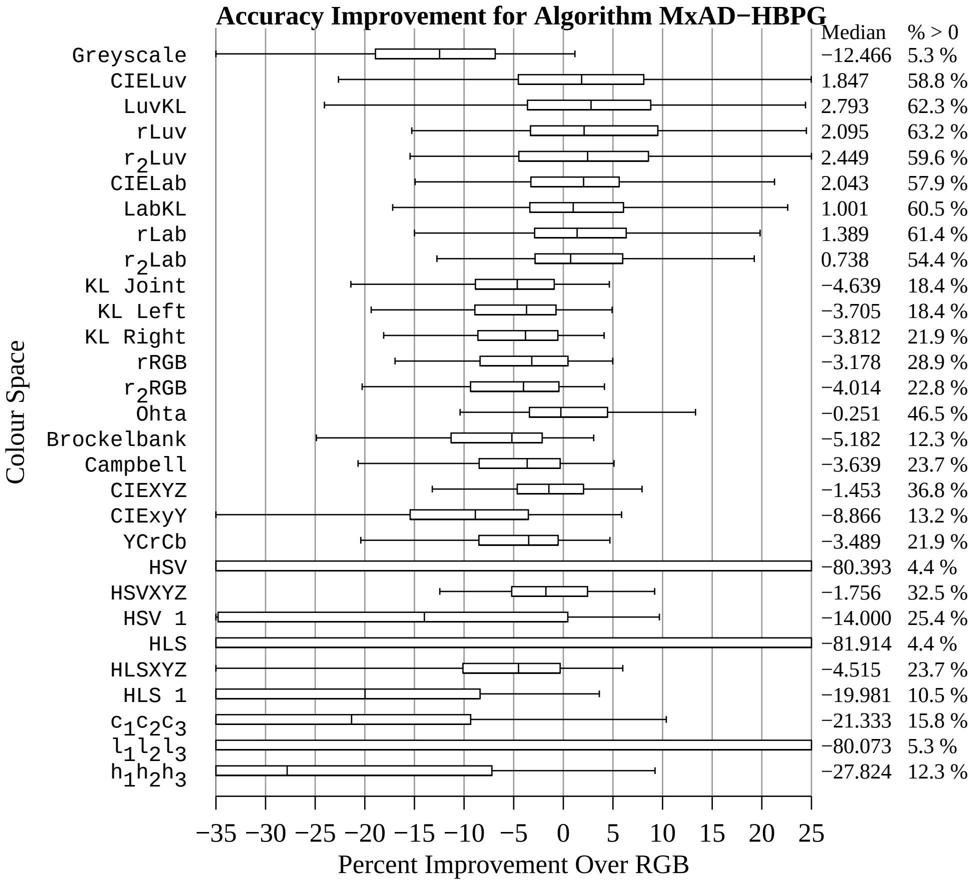


Figure K.61: Box plots of the over all aggregate percent accuracy improvement distributions for the MxAD-HBPG algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

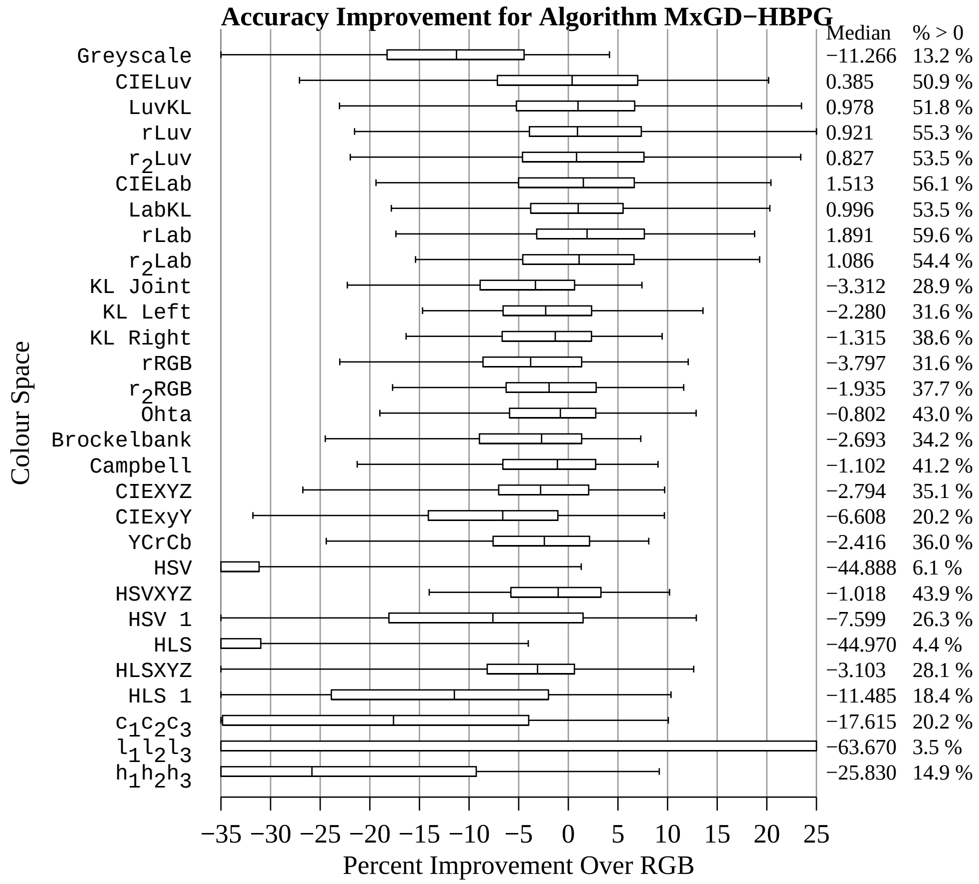


Figure K.62: Box plots of the over all aggregate percent accuracy improvement distributions for the MxGD-HBPG algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

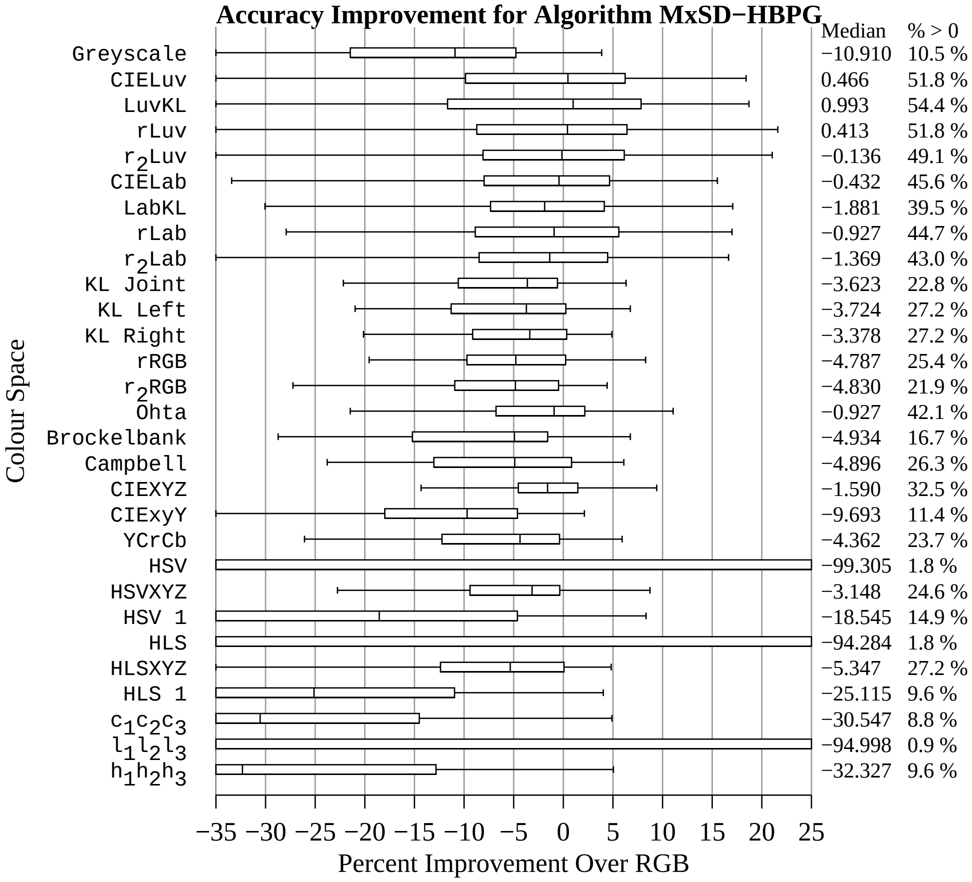


Figure K.63: Box plots of the over all aggregate percent accuracy improvement distributions for the MxSD-HBPG algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

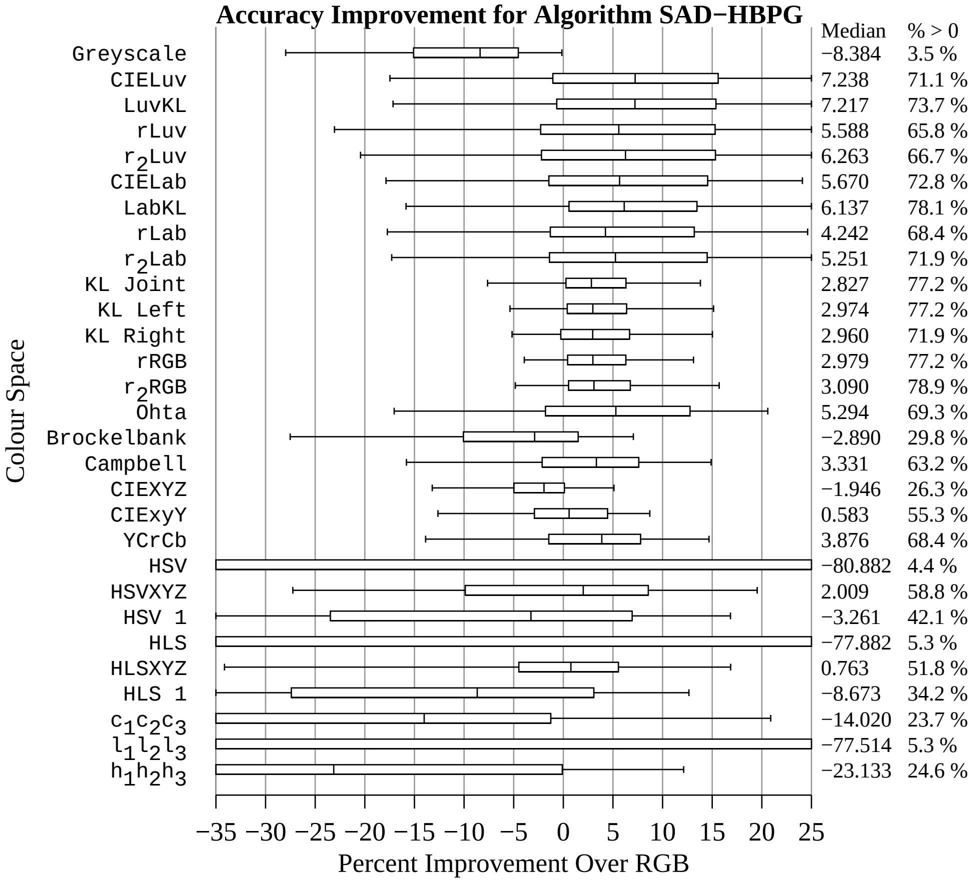


Figure K.64: Box plots of the over all aggregate percent accuracy improvement distributions for the SAD-HBPG algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

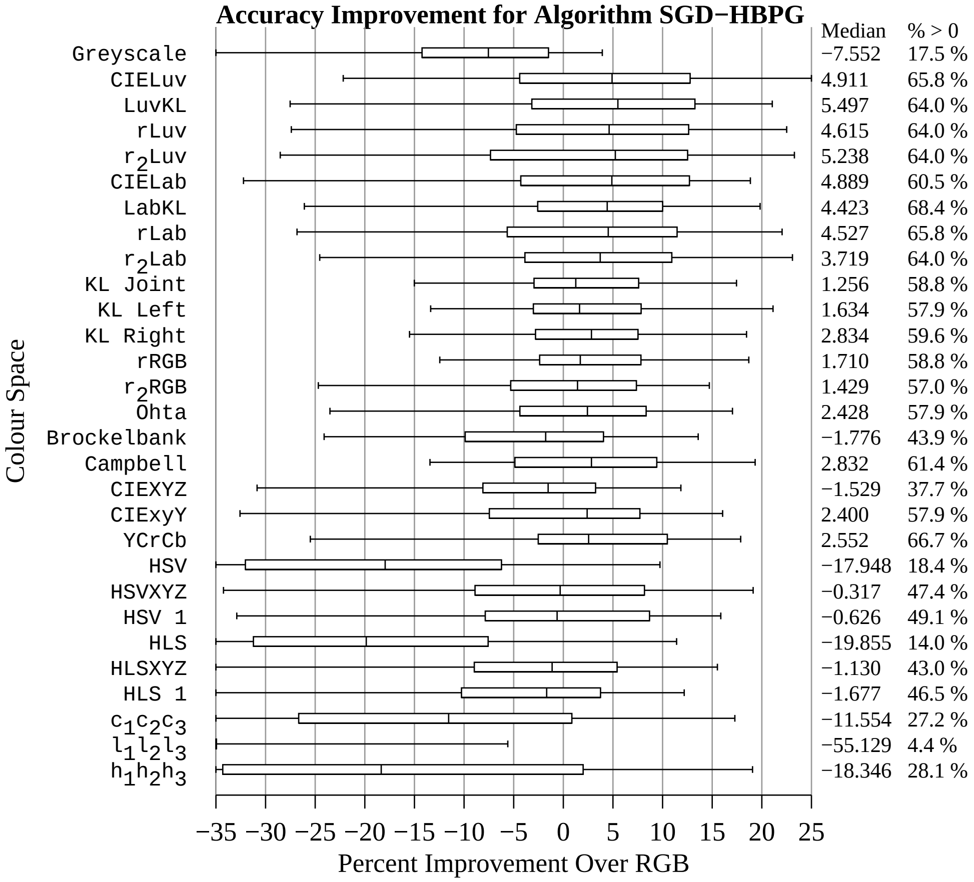


Figure K.65: Box plots of the over all aggregate percent accuracy improvement distributions for the SGD-HBPG algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

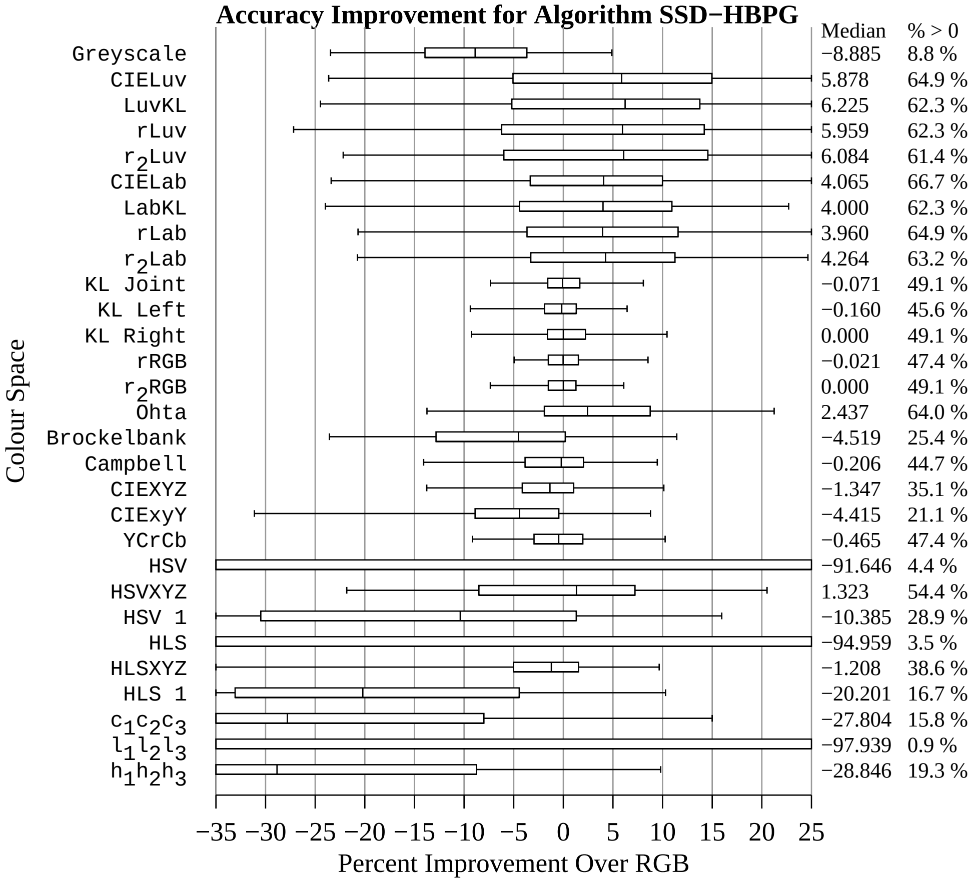


Figure K.66: Box plots of the over all aggregate percent accuracy improvement distributions for the SSD-HBPG algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

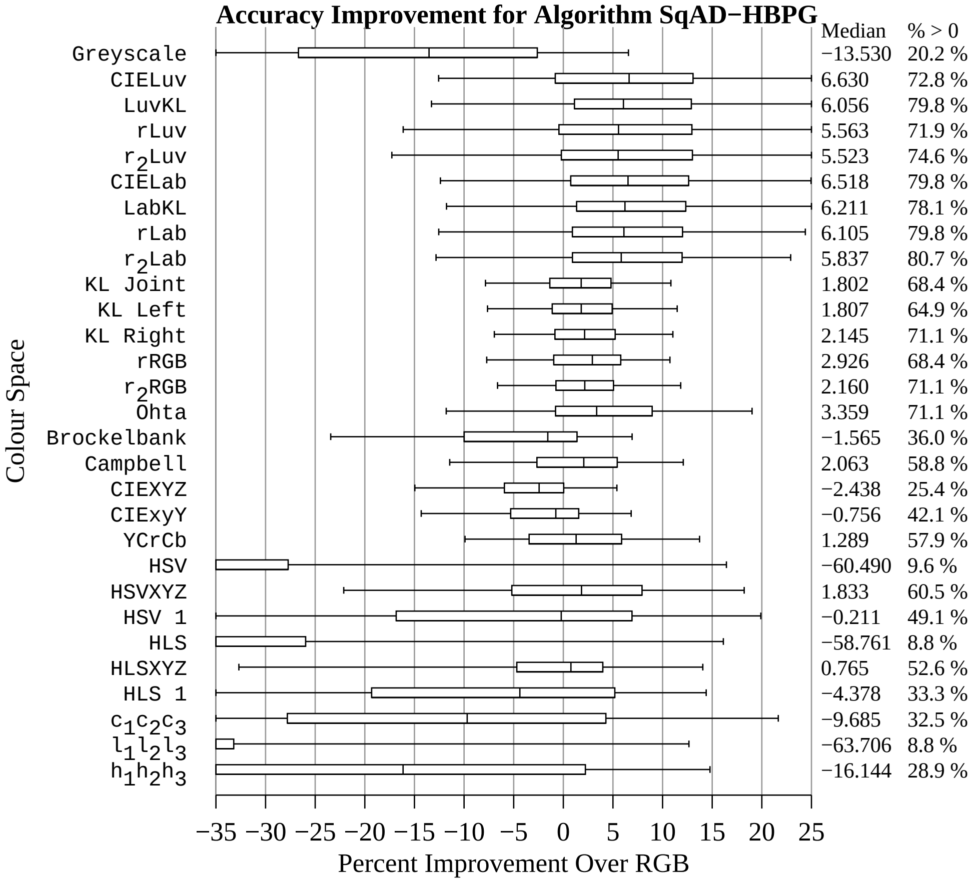


Figure K.67: Box plots of the over all aggregate percent accuracy improvement distributions for the SqAD-HBPG algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

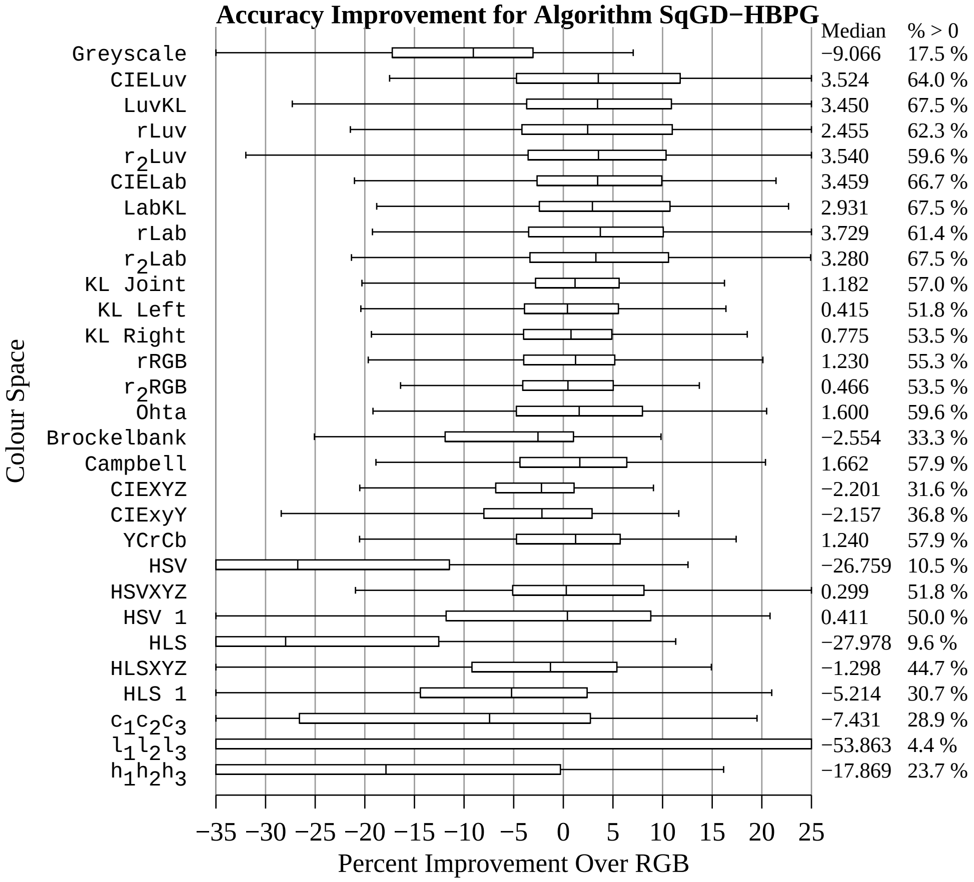


Figure K.68: Box plots of the over all aggregate percent accuracy improvement distributions for the SqGD-HBPG algorithm for each colour representation. The plots indicate the fifth, 25th, 50th, 75th, and 95th percentiles of the distribution.

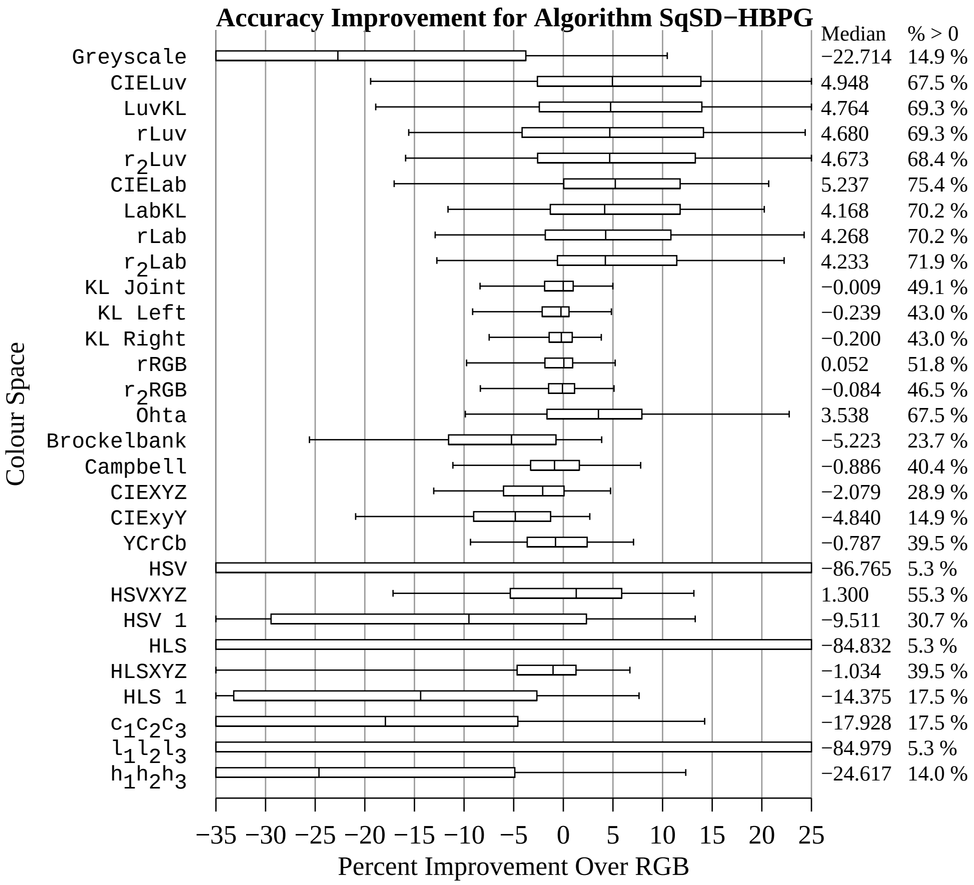


Figure K.69: Box plots of the over all aggregate percent accuracy improvement distributions for the SqSD-HBPG algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

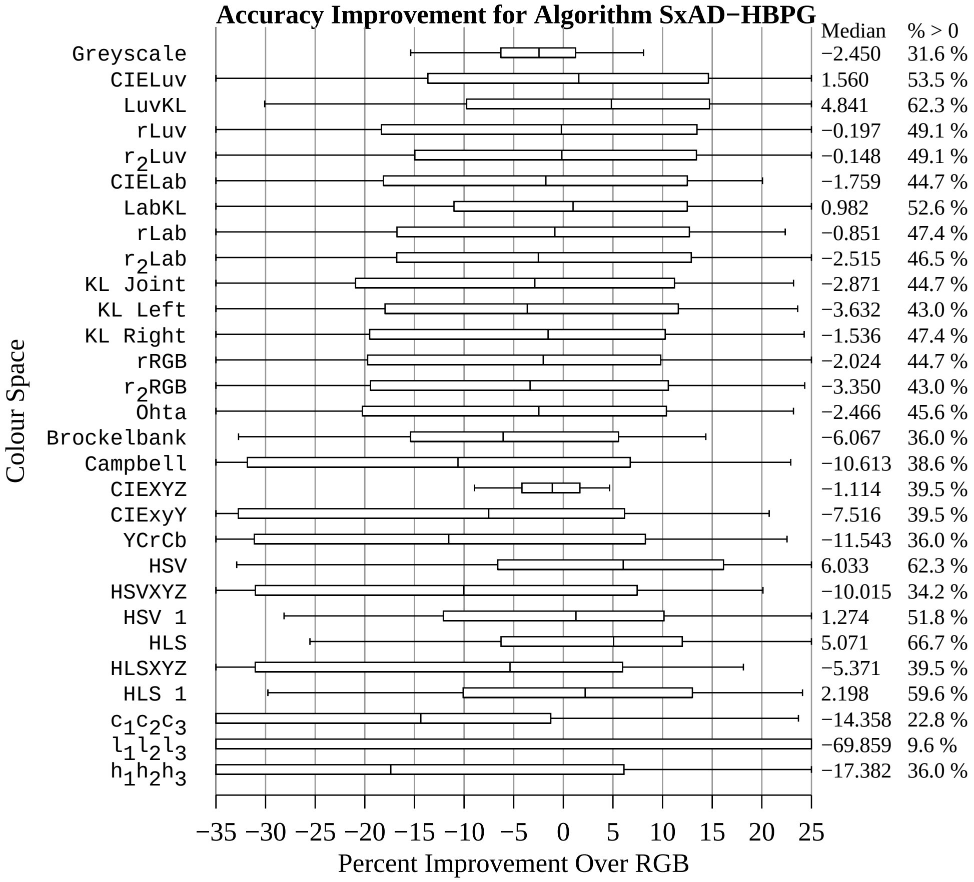


Figure K.70: Box plots of the over all aggregate percent accuracy improvement distributions for the SxAD-HBPG algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

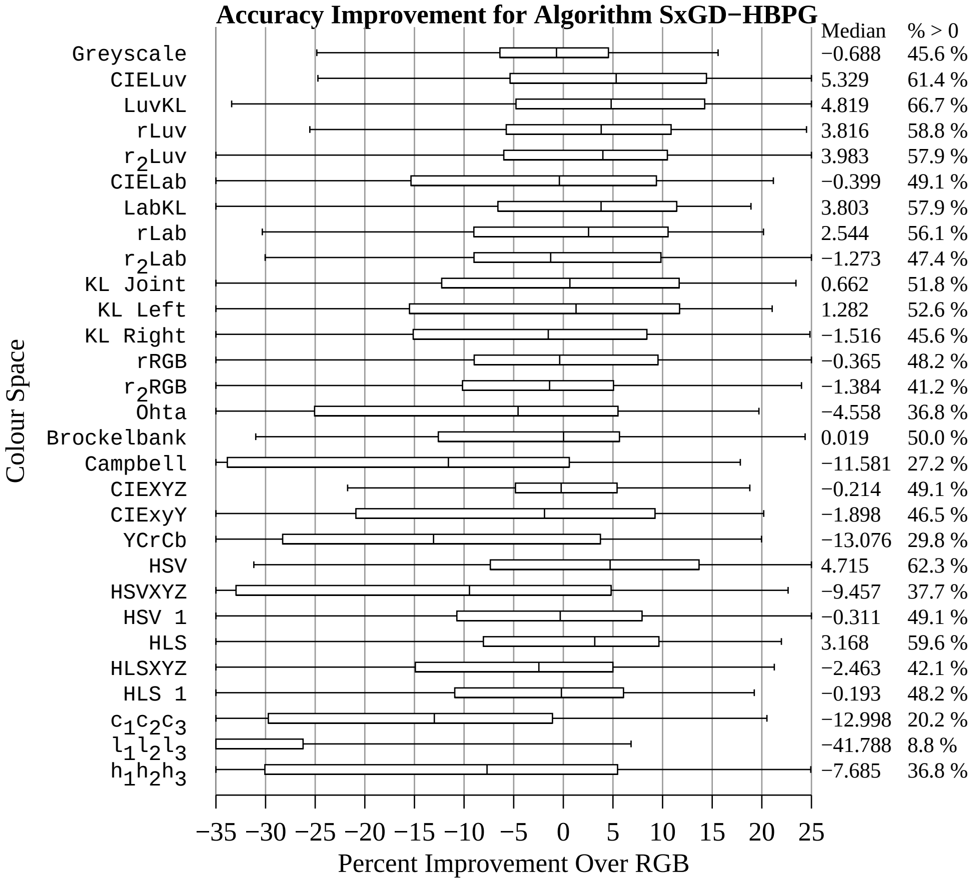


Figure K.71: Box plots of the over all aggregate percent accuracy improvement distributions for the SxGD-HBPG algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

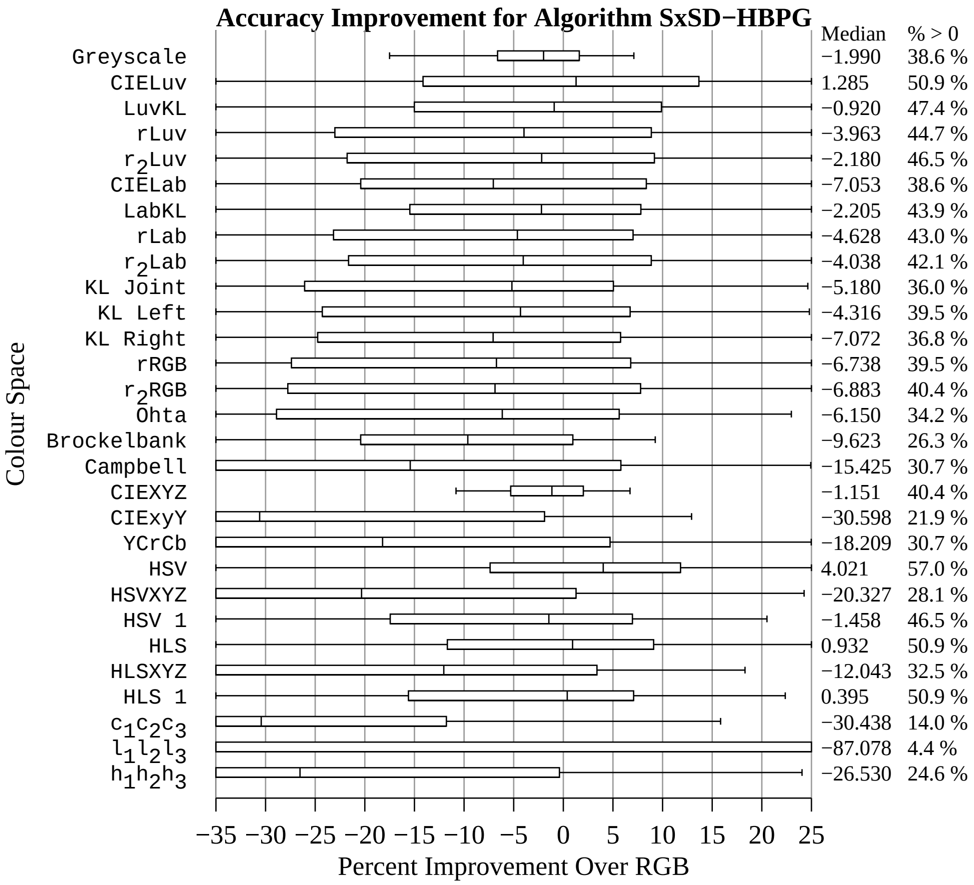


Figure K.72: Box plots of the over all aggregate percent accuracy improvement distributions for the SxSD-HBPG algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

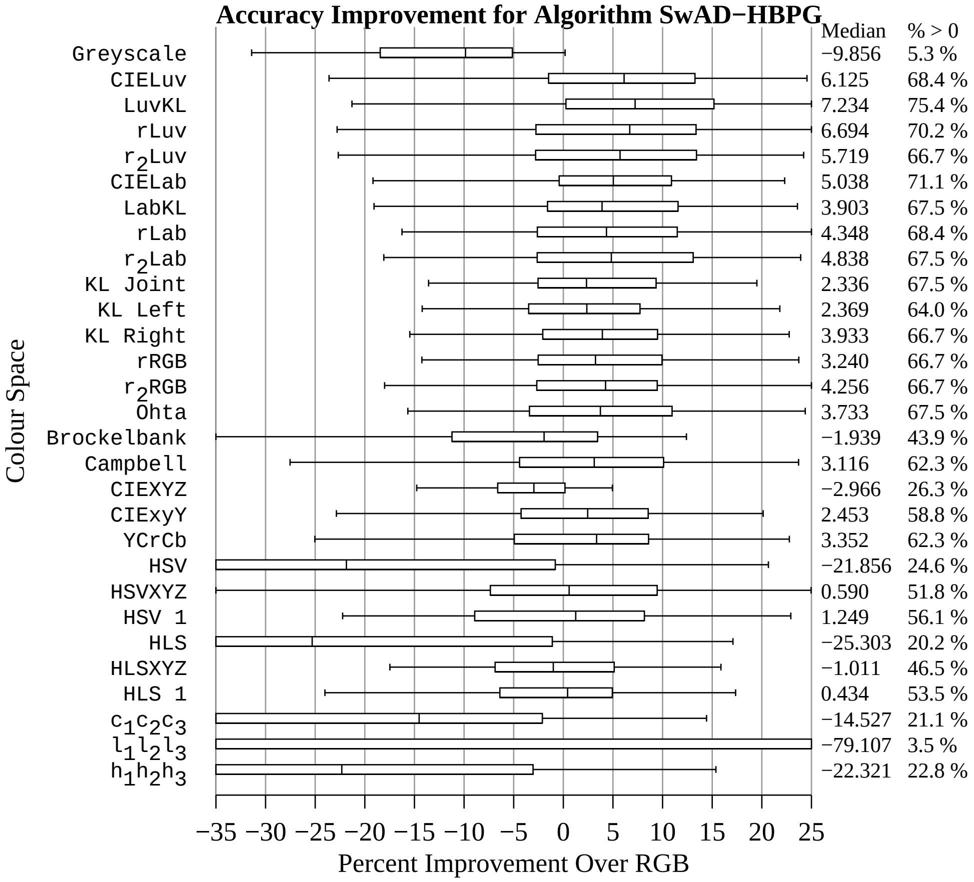


Figure K.73: Box plots of the over all aggregate percent accuracy improvement distributions for the SwAD-HBPG algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

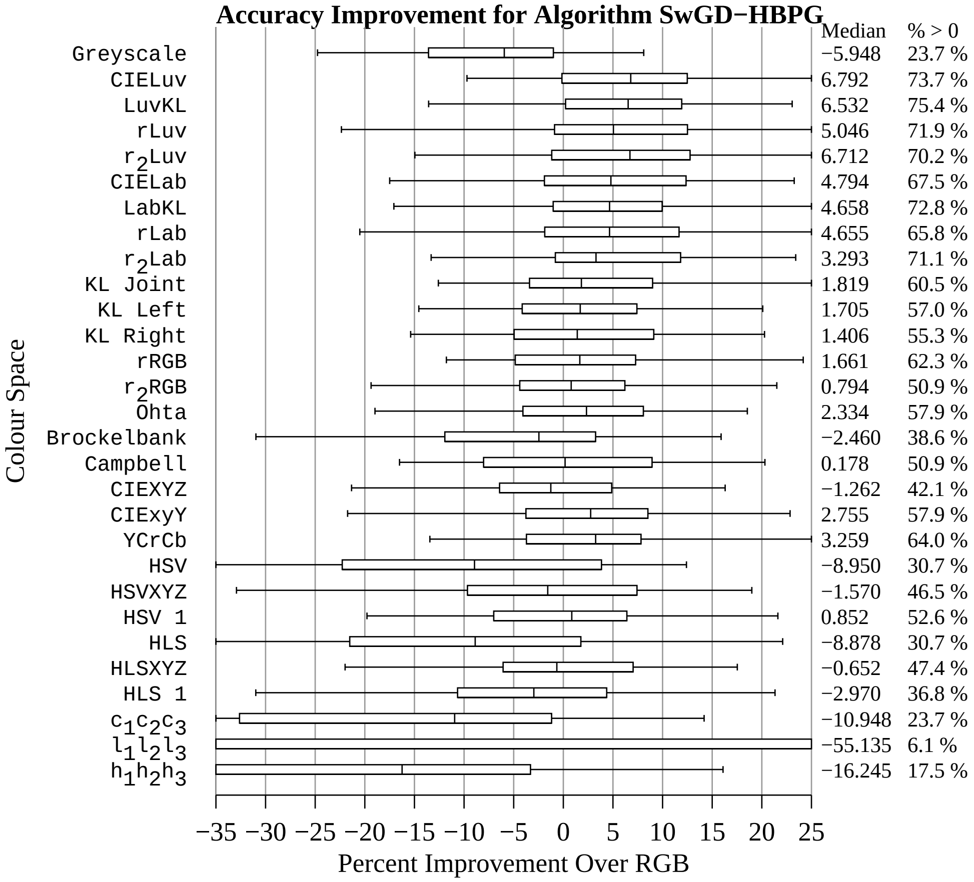


Figure K.74: Box plots of the over all aggregate percent accuracy improvement distributions for the SwGD-HBPG algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.

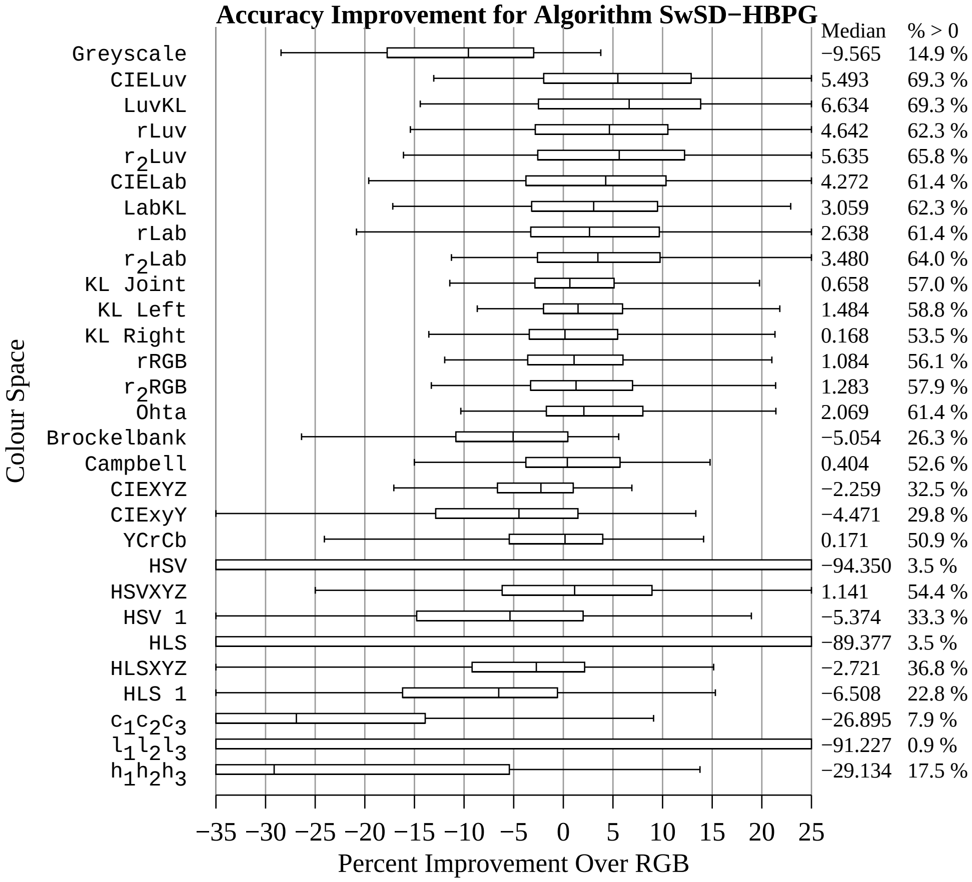


Figure K.75: Box plots of the over all aggregate percent accuracy improvement distributions for the SwSD-HBPG algorithm for each colour representation. The plots indicate the fifth, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentiles of the distribution.