

URBAN FOOD PRODUCTION: A PROTOTYPE DECISION SUPPORT SYSTEM

URBAN FOOD PRODUCTION: A PROTOTYPE DECISION SUPPORT SYSTEM

By

LAURA TOPPOZINI, B.Sc., M.Sc.

A Thesis

Submitted to the School of Graduate Studies

in Partial Fulfillment of the Requirements

for the Degree

Master of Applied Science

McMaster University

© Copyright by Laura Toppozini, April 2010

Master of Applied Science (2010)

McMaster University

(Civil Engineering)

Hamilton, Ontario

TITLE: Urban Food Production: A Prototype Decision Support System

AUTHOR: Laura Toppozini, B.Sc. (Lakehead University), M.Sc. (McMaster University)

SUPERVISOR: Dr. B.W. Baetz, P.Eng.

NUMBER OF PAGES: xi, 103

Table of Contents

List of Figures	vii
List of Tables	ix
List of Variables.....	ix
Abstract	xi
Acknowledgements.....	xii
Chapter 1 Introduction	1
Chapter 2 Literature Review	5
2.1 Introduction	5
2.2 Motivations.....	5
2.3 Programming Resources	11
2.4 Gardening Information.....	15
Chapter 3 DSS Development.....	18
3.1 Introduction	18
3.2 Conceptual Model	19
3.3 Shadow Analysis Documentation	21
3.3.1 Using Shadow Analysis	21
3.3.2 Code Description: Shadow Analysis	24
3.4 Garden Layout Documentation	27
3.4.1 Using Garden Layout.....	27
3.4.2 Code Description: Garden Layout	31
Chapter 4 Applications of the Developed DSS.....	45

4.1	Introduction	45
4.2	Hypothetical Application	45
4.2.1	Shadow Analysis.....	45
4.2.2	Backyard Garden	47
4.2.3	Front Yard Garden	54
4.2.4	Comments Regarding the Hypothetical Application	61
4.3	Actual Case Studies.....	62
4.3.1	DSS Results for User 1	62
4.3.2	DSS Results for User 2	67
4.3.3	DSS Results for User 3	71
4.3.4	Comments and Progress.....	74
Chapter 5 Conclusions and Recommendations for Future Work		79
5.1	Summary	79
5.2	Conclusions	80
5.3	Recommendations for Future Work.....	83
Chapter 6 References		86
Appendices.....		90
Appendix 1 User Instructions		90
A1.1	Introduction.....	90
A1.2	List of Useful Tools	91
A1.3	Shadow Analysis.....	92
A1.4	Garden Layout.....	96
Appendix 2 Hypothetical Trial Layouts		103

A2.1 Backyard Trials.....	103
A2.2 Front Yard Trials.....	103
Appendix 3 TotalPackage.rb Code	103
Appendix 4 Vegetable Database	103

List of Figures

Figure 3.1 Conceptual Flow Chart of Inputs and Outputs of the Developed DSS	20
Figure 3.2 Google Earth Zoomed to Include Relevant Features	22
Figure 3.3 Example Model to Show How Components are Used	25
Figure 3.4 Screen Shot to Show Shaded Model in SketchUp.....	26
Figure 3.5 Representative Rectangle with Edges and Face Selected in SketchUp.....	29
Figure 3.6 Screenshot of the Garden Layout Showing How/Where Quantities are Used	31
Figure 3.7 Diagram showing the angles angle1 and angle2	32
Figure 3.8 Diagram of the graphical layout showing the lengths of its elements.....	36
Figure 3.9 Scenario One: pairlay sorted in descending order of combined antagonist lists and name.	39
Figure 3.10 Scenario Two: pairlay sorted in descending order of combined antagonist lists.	40
Figure 3.11 Scenario Three: pairlay sorted in ascending order of combined antagonist lists and name.....	41
Figure 3.12 Scenario Four: pairlay sorted in random order.....	42
Figure 3.13 Scenario Five: pairlay sorted in random order a second time.	43
Figure 4.1 Model of Representative Parcel.....	46
Figure 4.2 Image After Shadow Analysis (May through August, 2010).....	47
Figure 4.3 Backyard Garden: Trial 1. Default settings for reaching length and path width (32” and 12”, respectively). This path width also coincides with the smallest path width.	51

Figure 4.4 Backyard Garden: Trial 2. Largest maximum reaching length (36") and default/smallest path width (12"). Notice that each pair is represented twice as many times as a single vegetable.....	52
Figure 4.5 Backyard Garden: Trial 6. Smallest maximum reaching length (12") and smallest path width (12"). Notice that the areas beside the paths are split into 3 separate areas.	53
Figure 4.6 Backyard Garden: Trial 11. Default maximum reaching length (32") and largest path width (30"). Notice that pairs are represented twice as many times as a single vegetable; this is considered preferable.	54
Figure 4.7 Front Yard Garden: Trial 1. Largest/default maximum reaching length and smallest/default path width (32" and 12", respectively).	58
Figure 4.8 Front Yard Garden: Trial 6. Smallest maximum reaching length (12") and smallest/default path width (12"). Notice that the areas beside the paths are split into 4 separate areas.	59
Figure 4.9 Front Yard Garden: Trial 11. Default maximum reaching length (32") and largest path width (30").	60
Figure 4.10 Front Yard Garden: Trial 19. Default maximum reaching length (32") and largest path width (30"). Fourteen vegetables were chosen to illustrate a limitation of the plugin; only twelve vegetables are present in the graphical layout. Two vegetables were not included because the number of paired vegetables in pairlist is more than the number of areas in the garden.	61
Figure 4.11 User 1: Shadow Analysis. The shadows represent the 24th of August, 2010.	64
Figure 4.12 User1: First Trial of Garden Layout.....	65
Figure 4.13 User 1: Second Trial of Garden Layout	66
Figure 4.14 User 2: Shadow Analysis representing August 24th, 2010.	68

Figure 4.15 User 2: First Garden Layout Trial.	69
Figure 4.16 User 2: Second Garden Layout Trial. Notice the areas beside the path are not divided into smaller areas.	70
Figure 4.17 User 3: First Garden Layout Trial.	72
Figure 4.18 User 3: Second Garden Layout Trial. Notice the relative dimensions; the length (shorter of the dimensions) is larger than the other trials.	73
Figure 4.19 User 3: Third Garden Layout Trial.....	74

List of Tables

Table 4.1 Backyard Garden Case Study	48
Table 4.2 Front Yard Case Study.....	55
Table 4.3 User 1's Input and Notes for DSS Trials	63
Table 4.4 User 2's Inputs and Notes for DSS Trials.....	67
Table 4.5 Details of User 3's Garden Layout Trials.	71
Table 4.6 Compilation of Actual Case Studies	75

List of Variables

alls: each element of this array contains the coordinates of each of the garden areas.

angle1: the angle between the x-axis and the foot of the garden

angle2: the angle between the x-axis and the left side of the garden. It is calculated by adding 90 degrees to angle1.

ant: a variable that represents the number of antagonists present in potential vegetables in an area that is adjacent to an vegetable-assigned area.

aunt: a hash of vegetables (from **pairlay**) assigned to the number of antagonists in the assigning of the vegetables to an area that is adjacent to an assigned area.

diffscen: a hash whose key is a hash of an assigned/planned garden layout and value is the number of adjacent antagonists in each plan. This will be used to choose the plan with the least antagonists.

edge: an array containing the vegetables that have been assigned to the garden whose area's edge is touching the edge of the area that is being planned.

pairlay: an array that contains **pairle** in different arrangements. This array is used for scenarios.

pairle: this array is **pairlist** multiplied so that the number of entries is equal to the number of areas in the garden.

pairlist: this array is **pairlisto** in reverse order; this increases the likelihood of all vegetables being included in the layout since **pairlist** is multiplied to equal the length of the number of areas in the garden.

pairlisto: an array of 2-d arrays (containing a vegetable pair or a vegetable and an empty hash) that is made after the companion suitability is checked. Pairs are entered twice into this array (at either ends) and the single vegetables are added to the end.

selout: an array of the selected entities in SketchUp.

vegarray: an array of vegetable hashes that were picked by the user in the input box. The antagonist and companion lists have been altered to include only the vegetables that were picked.

Abstract

Southern Ontario residents are faced with many challenging decisions when growing their own food. The intention of my research is to help these urban residents plan their garden plot in order to yield food for their own use. The form of this research will be a thesis incorporating a decision support system (DSS). This DSS is intended to take in and determine relevant site characteristics (latitude, sun/shade conditions) and use this information to help the user choose a variety of vegetables and herbs. Users will have the option of making a simplified model of their property and nearby structures for shade analysis, and with the results select an appropriate area(s) of their land. This DSS will give the user the freedom to pick vegetables based on conditions and preferences and give graphical and tabular output of the garden layout and details.

The objectives of this thesis is to present the why, what, who, where, and how of going beyond local food production for urban consumption to urban citizens growing their own food for themselves. This food can be consumed but also used as a currency with which to barter for other yard produce from neighbours or community members. One could imagine having a bartering relationship with a neighbour or having a weekly or monthly food market to facilitate bartering. This DSS is intended to be one of the building blocks of a food network DSS, which would be used to increase the efficiency of sharing food produced in urban residential gardens (that have been planned using the following DSS prototype).

Acknowledgements

I would like to thank my supervisor, Dr. Brian W. Baetz for all his guidance, inspiring ideas, and patience in the time before and while I've been his student, and, I'm sure, after as well. From the conception of this idea, I have kept a strong excitement and interest; due in part to the weekly brainstorming meetings with Dr. Baetz. My interest has persevered through the many twists, turn of creating and the different incarnations of the final product that is my thesis. I am truly proud to have worked under such an inspiring and influential man.

Thanks area also extended to McMaster's Department of Civil Engineering and the National Sciences and Engineering Research Council of Canada for their support of this research.

Chapter 1 Introduction

This thesis describes the underlying motivations and supporting concepts behind the development and application of an urban food production decision support system (DSS). A decision support system format was chosen in order to encourage the user (individual land owner, community group, etc.) to consider urban food production as a source of food to reduce reliance on conventional farming and increase productivity of the urban and suburban landscape. The concepts that motivate this DSS are the negative impacts that conventional farming methods are having on the environment and the positive impacts that urban farming and food networking have had on the surrounding community (Stonehouse, 2004; Waterloo, 2005; Howe 1999). Conventional large-scale farming creates large amounts of greenhouse gases (GHGs) in the cultivation and transport of food across long distances and the degradation and depletion of natural resources such as water, topsoil, agricultural lands, and forest and natural habitats. The positive impacts associated with urban vegetable gardens are the benefits resulting from access to fresh produce, frequent light exercise, an increase in community involvement, stimulation of the local economy with food selling partnerships, and an increase in the sustainability of the land (Howe, 1999; Beck, Quigley, Martin, 2003).

In addition, by decreasing a city's dependence on non-local sources of food by growing food locally and within the urban boundary, a city can be more resilient with respect to food in light of peak oil, climate change and economic fluctuations. A study based in Toronto, Ontario supports the assertion that community gardening encourages the community to become more involved in the local food security movement as well as to meet many needs of not only the individual gardener but the community as a whole (Baker, 2004). Currently in the United States, First Lady Michelle Obama has started a victory garden to bring a constant supply of fresh produce for the personal use of the presidential family and events hosted at the White House. This is the second food

producing garden on the grounds since Eleanor Roosevelt's victory garden during World War II (Djiang, 2009). In fact, many countries participated in the 'dig for victory' campaign during World War II and it was estimated that in 1944 one-tenth of food was grown in garden plots and allotments (Hough, 1995). In a similar vein, Russ Ohrt of Hamilton has started his own business, Backyard Harvest, whose focus is to provide ultra-local food to the community and build a stronger sense of community through food in the downtown core of Hamilton (R. Ohrt, personal communication, January 22, 2010). Food is grown on local residents' lots in exchange for part of the food grown there; the produce is then sold at the local farmer's market. Ohrt has a long history with food production, including biodynamic and organic farm training which he incorporates into his business practices. Since Backyard Harvest started, in 2009, it has increased its land base three-fold and will start a community supported agriculture (CSA)-based structure where customers will purchase an amount of food prior to the growing season and have food available on a weekly basis.

The scope of this DSS is intended for urban and suburban residents interested in growing edible flowers, herbs, and vegetables as well as community groups looking to grow food to create an urban food network where food can be traded or sold within that community. The future scope could be extended to rural counterparts. The geographical region of interest is specifically Hamilton and southern Ontario, but more generally, Canada and areas with similar climate zones and food needs. It could be conceived that, for individuals and groups in different climates with different food requirements than the ones laid out in the decision support system, another database containing the relevant food information could be substituted for the current database to provide the same service.

This urban food production decision support system is comprised of two modules with details on how the code was written and instructions on how to use the modules for the user. Scripts were written in Ruby (Google, 2009a), to be downloaded and used with Google SketchUp (Google, 2010a), a 3d modelling software. Google Earth (Google, 2010b), a software utilizing satellite images of Earth, and GIMP (The GIMP Team,

2009), an image manipulation software, are also required for this DSS. These three programs were chosen as the basis for this DSS because they are all available as either freeware or open-source projects (all free to download and use), making them more accessible to the general public, and an alternative to pay-for-use, proprietary garden planning software. The first module, ShadowAnalysis, is designed to assist the user in modelling a three dimensional representation of their property and all relevant structures that contribute to shadows on the property, display a series of shadows, and analyze the property to help the user in selecting appropriate areas for the garden(s). Both Google Earth and Google SketchUp are used to capture a Google Earth image into SketchUp to act as a guide for the 3-d representation. Some simple depictions are available to aid the user in this process: a fence, house (with a triangular prism rooftop), cube (for depicting apartment buildings, warehouses, etc.), and two types of 2-d trees (coniferous and deciduous). After the representation is complete, the ShadowAnalysis script can be run, which will generate many scenes with representative light conditions throughout a specified time period. These scenes will be exported from a SketchUp file (.skp) to a JPEG (.jpg) and saved to the desktop of the computer. Using GIMP, these time-lapse images can be combined to allow the user to observe shadow patterns and gain a better understanding of where to place the garden(s).

With this information in hand, the user is prepared for the second module, Garden Layout. The user is required to create a rectangular face on the x-y plane that will depict the garden in size and location and, pending some selections made to the rectangle, activate the script Garden Layout. The user will specify various preferred dimensions as well as select a number of vegetables to be placed within the garden. A graphical layout that has been determined based on the companion and antagonist relationships of the vegetables then appears with colour-coded areas that designate rows of vegetables with appropriate spacing; this can then be used to prepare the garden. Additional resources such as a database of all vegetables can be used as a reference to the user throughout the growing season in order to predict conditions such as harvest dates, gardening tips, etc.

Through this decision support system, the objective is not only to help and encourage the users to produce some of their own food consumption, but to motivate this behaviour on a larger scale. This DSS is intended to be part of a larger food networking decision support system to aid in the overall goal of creating more sustainable and self-sufficient cities and suburban areas.

This thesis is structured as follows: Chapter 2 contains a literature review outlining sources used to explain the motivations for this thesis, programming within the various software packages, and gardening techniques; Chapter 3 contains a conceptual model of the decision support system, describing the DSS through a flow chart, and a description of the development of the DSS, by providing an in-depth and technical description of the DSS that focuses on functional and programming aspects. Chapter 4 contains information and results from the applications of this DSS, for a hypothetical case as well as actual test cases. Chapter 5 provides an overview of the thesis, draws conclusions from this research, and makes recommendations for further research pertaining to this specific DSS and this general field of interest. References and appendices follow.

Chapter 2 Literature Review

2.1 Introduction

There are three sections for this literature review, which relate to the necessary background for this research: motivations for the decision support system (DSS), programming resources, and gardening information. Motivations for this urban food production DSS are centred around conventional farming and its negative impacts, local and organic farming's improvement of these impacts, and the positive impacts of community-based farming, selling, and networking on the community as a whole. The softwares chosen for this DSS are Google Earth, Google SketchUp and GIMP, and a major contributor to programming resources are manuals and programming guides for Ruby, a scripting language used in Google SketchUp. Gardening information consists of growing conditions for, and detailed information about edible vegetables, herbs, and flowers that can be grown in a Canadian climate, but also considerations about garden layouts in terms of space, light, and the plants within.

The format of this literature review will be such that every section has an introduction and the subsequent information will explain each resource by outlining the knowledge gathered in each case.

2.2 Motivations

The motivation for developing a decision support system (DSS) rooted in urban agriculture is to encourage the productive use of the urban and suburban landscape for food production. Increasing the number of individual households that utilize yards and community gardens as a source of food can offset the cost of purchasing fresh produce, provide access to fresh produce, increase the sustainability of previously grassed land,

and attenuate the negative impacts associated with conventional farming. Local and organic farms support the reduction of these negative impacts (Waterloo, 2005; Weber, 2008; Hopp and Gussow, 2009), and by encouraging urban food production, it can be hypothesized that this support would be strengthened as well as introduce more benefits to the community. These impacts affect the community in several ways, including socially, economically and environmentally (Howe and Wheeler, 1999). The purpose of this section is to convey the motivation behind this decision support system that gives rise to the following thesis.

The publication by Howe and Wheeler (1999) is the first to be mentioned due to its broad yet detailed case for urban food gardens and their utility on a variety of levels. The authors identify environmental, social, economic, education, and health factors as the areas in which urban food gardens have positive impacts and compare two cities, Leeds and Bradford in the United Kingdom, to provide detailed, region-specific findings. Howe and Wheeler (1999) specifically perform an analysis of allotments, urban farms, and community gardens in Bradford and Leeds, but recognize that the scope of urban food growing includes: allotments, urban farms, community gardens, local authority tenanted farms, food grown in gardens, yards, etc., food grown in schoolyards; and food grown in prison grounds. This thesis uses this article to infer that the range of urban food growing opportunities reap benefits in much the same way that is shown for allotments, urban farms, and community gardens.

For each general area, the authors outline specific advantages that are provided by urban food growing. On the environmental front, it serves to reduce waste and transportation, and increases biodiversity by reducing chemical inputs. Social benefits are seen to be urban renewal and fighting both crime and discrimination by encouraging community engagement, interaction, and integration. The region hosting the urban food growing has the opportunity to benefit economically by supporting vocational training, and partnerships between urban food growers, sellers, buyers and other organizations to help produce goods and services in the local economy. Education can take the form of involving students in activities in a school yard garden or creating educational activities

on urban farms. And finally, urban food growing can provide physical benefits by giving citizens access to fresh fruit and vegetables as well as providing light exercise for tenders of the garden (Howe & Wheeler, 1999). Gardening has also long been used in the treatment of mental illnesses (Howe & Wheeler, 1999). This listing of the benefits of urban food production is an obvious resource and this paper is, therefore, an invaluable asset to the motivation of this thesis.

Baker (2004) gathers information on community gardens in Toronto, Ontario, specifically Francis Beavis Community Garden, Shamba Community Garden, and Riverside Community Garden. This study argues that community food gardens encourage its gardeners to become part of the local food security movement and that community gardening meets many needs of both community and gardener. These needs that are met include, but are not limited to, income offsetting, recreation, nutrition supplementing, community development interest, and social connections (Baker, 2004). This study serves to argue the case for community-based urban food production and the benefits of gardening with respect to individuals and communities in an urban center in southern Ontario.

Mendes, Balmer, Kaethler, and Rhoads (2008) raise the issue of urban planners and their lack of involvement in urban agriculture initiatives in their cities. This paper shows many reasons for this gap and looks at two cities (Portland, Oregon, U.S.A. and Vancouver, British Columbia, Canada) to compare their identification of public lands suitable for urban agriculture within their land inventories. The purpose of the publication is to find if the land inventories can be used to facilitate the integration of urban agriculture into planning and policymaking as well as to find if these land inventories progress the social and environmental aspects of the city's sustainability plan (Mendes, W., Balmer, K., Kaethler, T., Rhoads, A., 2008). In the past, planners have not attempted to plan for urban food production due to factors such as the perceptions that: food systems are a private (rather than public) sector and/or rural (rather than urban) issue, urban agriculture is not in their realm of responsibilities, and/or a lack of knowledge in the subject of urban agriculture (Mendes, W. et al., 2008). This paper shows that planners

are changing their scope more and more to include food systems. Mendes et al. (2008) found that land inventories were successful in facilitating the integration of urban agriculture into planning and policymaking in both Portland and Vancouver but that only Portland advanced their environmental and social aspects of their sustainability agenda, whereas Vancouver's small scope did not affect this change.

Stonehouse (2004) highlights some of the issues concerning the agriculture-food sector in Ontario. The main problems include decreasing amounts of farmland actually being used for farming and increasingly mechanized processes and synthetic inputs on the farms that do exist. Ontario's agricultural lands, that could conceivably be used to feed the Ontario population, are being minimized by urban-industrial complexes (Stonehouse, 2004). If this type of situation is continued it is easy to see that the amount of food that needs to be imported to the province will also increase, decreasing Ontario's ability to be self-sufficient with respect to food. Suggestions to improve these sustainability problems are to make a shift to reduced-input farming or organic agriculture practices and the development of policies to encourage and incentivize conservation efforts on the part of farmers. This paper addresses the problems of resource depletion and allocation within southern Ontario and the Great Lakes Basin, and it shows that the general planning province-wide lacks a cohesive strategy to be sustainable with respect to food production (Stonehouse, 2004).

Hough (1995) argues that urban growth causes a decline in rural areas surrounding the urban core, thus reducing the ability for these rural counterparts to provide agricultural services to the adjacent city. An example of this occurred in the early 1980s when Ontario lost over 4,000 hectares of agricultural land to urban development (Hough, 1995, p.7). During World War II, the 'dig for victory' campaign prompted many countries to grow fruit and vegetables in or near urban centers; it was estimated that during 1944, 10% of food was grown in garden plots and allotments (Hough, 1995). In China, policies have been in place to ensure urban municipalities are the main producers of their own food. For instance, Shanghai and Beijing are self-sustaining with respect to vegetables, and China has more than 85% of vegetables consumed within cities produced

in their respective municipalities, while meat products are usually raised on the outskirts of urban centers to minimize transport (Hough, 1995, p.214). Thus, it can be seen that even within densely populated cities (such as Shanghai or Beijing) urban food production is feasible and has been accomplished.

Emergy (embodied energy) is defined as "an expression, in one type of energy (solar energy), of all the available energy used directly or indirectly in the production of a product or service" (Beck et al., 2003). Beck et al.'s (2003) study, conducted in Columbus, Ohio, compares the amount of solar energy (every energy quantity is expressed in terms of solar energy) that is input and output from four different types of gardens. The garden types studied were conventional ornamental, edible ornamental, intensive organic, and forest gardens; all were grown in 9m by 6m lots (Beck et al., 2003). The outcome of this study revealed that all three of the non-conventional yards had emergy yield ratios (the emergy of the yield of the garden divided by the emergy of the material and service inputs) that were three orders of magnitude higher than the conventional yard. An emergy yield ratio of one suggests that the embodied energy input into the system is equal to the output; ratios less or greater than one mean that the economy is at a disadvantage or an advantage, respectively. None of the garden types studied had a ratio of larger than one but the general outcome of the ratios showed that the intensive organic, edible ornamental, and forest gardens yielded ratios of the same order of magnitude whereas the conventional ornamental garden had an emergy yield ratio that was three orders of magnitude lower than the other gardens. Though all gardens had ratios much less than one, a large portion of the input emergy (the denominator of the ratio) was the labour invested into the maintenance of the yards (Beck et al., 2003). One might argue this expenditure to be a benefit to the gardener, decreasing the input, and thus increasing the emergy yield ratio.

This analysis was done for the initial year and projected to five years with similar results; all emergy yield ratios increased as a function of time in somewhat the same proportion. Another point to note was the sustainability index (a ratio of the emergy yield ratio and the environmental loading ratio (economic plus non-renewable inputs over

renewable resources)) achieved in the cases. A sustainability index above 1.0 is an indicator of a sustainable system, and in a garden setting, indicates the garden is having a net positive effect on the environment. All of the gardens studied had a sustainability index *below* unity; however the three food garden configurations were much more sustainable than the conventional garden (the food gardens had sustainability indices three orders of magnitude larger than the conventional garden) (Beck et al., 2003). This study serves to show that these types of garden configurations are more sustainable than the way conventional urban/suburban yards are used.

The Public Health Department in the Region of Waterloo has released a report that assesses food imports that are indicative of food preferences and foods that can be produced in the Waterloo Region and compares the environmental impacts of transporting these foods against sourcing the same products locally. In their study, within the top ten contributors of greenhouse gas emissions are pears, lettuce, tomatoes, potatoes, bell peppers, apples, onions, and carrots. Of the foods surveyed, the average imported distance was almost 4,500 km, generating 1.3kg of greenhouse gasses (GHGs) per 1kg of food; food sourced from southern Ontario travels an average of 250 km, generating 0.067 kg of GHGs per 1kg of food transported; and finally, food sourced from the Region of Waterloo travels an average of 30km and generates 0.008kg of GHGs per 1kg of food transported (Waterloo, 2005). This shows that when foods that can be grown locally are purchased locally, this can have a dramatic effect on the amount of GHGs produced. One can infer that producing vegetables in urban cores, where most people reside, could reduce GHGs further.

Another food miles study conducted by Weber (2008), analyzed the climate impacts of various categories of food, including fruits/vegetables and cereals/carbohydrates. They argue that greenhouse gasses can be reduced by consuming less red meat and dairy and buying more local food. Hopp and Gussow (2009) comment that buying local food reduces greenhouse gasses, not only from the aspect of transport, but from the likelihood of the practices of local food production being organic and the waste being minimized. From these publications, one can ascertain that by producing

vegetables in urban centers, the benefits of reduction of GHGs from transport, reduction of waste products, etc. are increased more so than sourcing these foods from local rural areas.

2.3 Programming Resources

In order to make the developed prototype decision support system available to the public free of charge (provided the user has a functioning computer with appropriate space and an internet connection for downloading the applications and amendments), free versions of Google Earth, a satellite image resource; Google SketchUp, a 3D modelling software; and GIMP, an image manipulation program, were used. Ruby is the programming language used within Google SketchUp and therefore, the majority of resources used to create this DSS are general Ruby programming resources. Other decision support systems that are specific to small-scale gardening and farming are also discussed in order to set the stage for this DSS and to give background on DSSs that are available to gardeners currently.

The Google SketchUp Ruby API is a resource that provides commands to use in programming with Ruby that are specific to Google SketchUp. The Ruby application program interface (API) within SketchUp allows the user to create scripts written in Ruby that can then be saved (in specific files) which are then loaded in SketchUp. This web page also provides useful tutorials, tips on getting started, and a quick reference to classes, methods, and objects (Google, 2009a). Because Ruby is an object-oriented programming language, the object reference, which contains commands within different object classes, was of major importance to the programming portion of this decision support system. Also within this resource, commands concerning Google Earth were found (Google, 2009a). Minimal programming using Google Earth commands was needed in SketchUp for this DSS, but it should be noted that no other references were used for Google Earth in this sense.

SketchUcation.com is a website that is devoted to providing information for the users of Google SketchUp. This resource provides its information from people from many different backgrounds (for example architecture, engineering, woodworking, landscaping, design and more) in the form of tutorials for various stages of knowledge, extensions (for users to purchase or use for free), and forums for a variety of uses, a variety of skill sets and a variety of issues or topics (Google, 2009b). Within the Community Forums page, the Plugins Forum and Developers' Forum were the main sources of information for the writing of this DSS. I received responses to questions regarding general programming information in Ruby as well as advanced topics that are very specific to SketchUp's object classes. Information in the forum is discussed by many levels of users of the forum.

Ruby is an established programming language and as such, it has many books to be used as teaching tools and references for those using Ruby. The two main books that were used to understand general information, functions, and commands were *Learning Ruby* (Fitzgerald, 2007) and *The Ruby Way* (Fulton, 2002). *Learning Ruby* is written for, as the title may imply, people learning to program in Ruby. It gives an overview of all types of methods and classes and examples clearly showing their uses. This book helped create a framework on which to build the entire programming portion of the developed decision support system. *Learning Ruby* and *The Ruby Way* also acted as reference guides during the programming process of the DSS for general programming; and though neither contain any reference to Google SketchUp, both books provided a wealth of knowledge for Ruby.

Version 2.6 of GIMP (GNU Image Manipulation Program) is the intended version of GIMP for the user to analyze the image files exported from SketchUp using the ShadowAnalysis plugin. All information gathered about the software that is not immediately apparent by using the interface itself was retrieved from the user manual documentation provided online by The GIMP Documentation Team (2009). An example of this is the use of the layers, which is needed for the shadow analysis, and how the settings can be changed with respect to each layer. The function of linking all layers

together in order to change their properties simultaneously instead of having to change the settings one at a time (which can be tedious and unpleasant if the user chooses to create many layers) is not currently a function that exists in GIMP, but this manual may be a starting point for writing a script to accomplish this task. This program is a helpful tool for the user in planning where to place gardens and the lighting conditions of the proposed gardens.

In order to plan the form and functionality of this decision support system prototype, knowledge of current tools and applications of similar function was needed. Many tools of potentially similar function were inaccessible due to cost. The programs reviewed for information on available services are outlined. The following three internet-based garden planning tools are the most similar gardening programs to the developed DSS and were used as a starting point in order to improve on the features provided. Growveg.com, Plangarden, and Gardener's Supply Kitchen Garden Planner are all accessible with an internet connection directly through an internet browser; Plangarden also has the option of downloading the software to one's own computer (Plangarden, 2010). In both Growveg.com and Plangarden the user has the ability to make various shapes and sizes of gardens where the 'drag and drop' feature combined with a visual buffering zone allows the user to easily place each vegetable in a position that accounts for plant spacing. These two programs also have the added feature of extending the number of plants to accommodate a row by clicking and dragging the vegetable to the length specified by the user (Growing Interactive, 2010; Plangarden, 2010). Gardener's Supply Kitchen Garden Planner (KGP) allows only rectangular-shaped gardens and uses dropdown menus in feet for the user to select the length and width (Gardener's Supply Company, n.d.). The gardens are laid out such that square foot partitions are visible and each square has one type of vegetable assigned to it by the user. KGP also has a 'drag and drop' feature, though instead of having a visual buffer, when the vegetable is dropped into one of the squares, the software automatically displays the number of plants that can fit within the squares according to space requirements. Though each of these products provides information on companion planting, Growveg.com is the only one to

incorporate it into the software. Growveg.com allows the user make gardens for multiple years and depending on what was planted previously, a red area will appear when placing the vegetable to alert the user that a vegetable in that same family was planted there previously. This can help problems like nutrient depletion and/or reoccurrence of soil diseases or pests (Growing Interactive, 2010).

For each of these programs information about the vegetables chosen can be found in tabular form and details of sowing, spacing, harvesting and tips on growing are available. Growveg.com gives a graphical chart of sowing and harvesting times for all chosen vegetables. These garden softwares are designed for individual gardening endeavours and Growveg.com and Plangarden can be purchased for \$25 USD (Growing Interactive, 2010) and \$19.95 USD (Plangarden, 2010); while KGP is free (Gardener's Supply Company, n.d.).

I-farmtools.org is a free, internet-based application from Iowa State University and is intended for farmers and decision makers as an integrated crop and livestock production and biomass planning tool. This decision support system uses soil and weather data from all US states and is available for different skill levels. The user can select an assortment of crops and crop rotations using a variety of farming practices, such as tillage, fertilization, planting, weed control, harvesting, and residue removal. The output of this tool is a number of tables including information about crops, forages, and biomass imports and exports from the farm; manure and fertilizer imports and exports from the farm; nutrient balances (N, P, and K) at the field and farm scale, and economic impacts at the farm scale. I-farm also has an optional field-drawing and data-finding GIS tool where the user can draw a 'field' represented by a polygon and information such as soil-type, slope, and other attributes are conveyed to the user. This program is a tool designed for farm systems and therefore not appropriate for an urban environment, but does incorporate a similar, though more high-level, vision that can act as inspiration for less complex systems like small scale food gardens (I-Farm Team, 2010).

2.4 Gardening Information

Embedded within this decision support system is information about edible vegetables that is an essential part of the functioning of the DSS. The information gathered has to be somewhat technical in order to represent the vegetables appropriately and be relevant to Canadian gardeners. Information was sought out, including a broad range of knowledge with respect to gardening and computer knowledge, such that any general user will be able to find this decision support system easy to use and useful for their purposes.

Beck (2008) gives an overview of the many edible vegetables, herbs, flowers, etc. that can be grown in the various Canadian climates. In this publication, each vegetable has its own section with the headings: Starting, Growing, Harvesting, Tips, Recommended, and Problems and Pests. These sections, therefore, give the reader a detailed overview of what to expect in the cultivation of these vegetables which provided much of the information for the plant database in Microsoft Excel, including dimensions of fully formed plants, preferable soil conditions, as well as a vegetable's aesthetic properties. Because vegetable spacing is not included explicitly for each plant listed, half of the average of the spread of each plant (except where an explicit spacing was listed in the growing tips) is included in the VegetableDatabase.xls. Many attributes within this book are included within the Garden Layout's database in order to ensure a representative layout of the garden plot.

The New Northern Gardener (Bennett, 1996) is a resource for gardeners and gardens in Canada and the northern United States. Gardening information in the form of garden design strategies were modified from Bennett (1996). Four different vegetable garden layout types were described and from this, the garden layout is partially based on the temporary and permanent wide-row gardens. Depending on the preference of the user, the garden can incorporate temporary or permanent features. The list of vegetable yields and vegetable space efficiency chart were resources used in the consideration of the vegetable list. Bennett (1996) also has a compilation of both vegetables and flowers

(edible and non-edible) that were used to consider the different types of vegetables, herbs, and flowers to enter into the database.

Bomford (2009a) specifically refers to a specific type of agricultural method, biointensive agriculture, in an analysis of land-use efficiency for three plants which have different relationships in biointensive agriculture. The plants used in this analysis were tomatoes, basil, and Brussels sprouts and, as the name of this paper would imply, planting tomatoes and basil together is thought to be beneficial to the growth of tomato plants and Brussels sprouts are said to be detrimental to tomato growth in the biointensive regime (Bomford, 2009). Over the two years, 2001 and 2002, in which the experiments were done, weather contributed to results of an inconsistent nature and no conclusive information confirming or disproving the validity of these pairings was confirmed. The land-use efficiency analysis, on the other hand, resulted in recommendations by the author of methods to space the mixed plants in a more efficient manner. This recommendation is accompanied with a Microsoft Excel spreadsheet (Bomford, 2009b) that can be used for the pairing determination if the user of this DSS wishes to alter the layout for each vegetable pair.

Jolliffe (1997) synthesizes information from over fifty published experiments in order to create a review of the many two-species mixtures compared to pure stands of single species. Hundreds of direct observations and estimates are used to make plots of number of cases vs. relative land output. Relative land output (RLO) was the main analysis quantity used within this compilation; RLO "involves equivalent total density, individuals per species, and land area in mixed and pure stands" (Jolliffe, 1997). Jolliffe (1997) uses this quantity when comparing the biomass produced by each species grown by way of two-species mixed and pure stands. Overall, it was determined in this publication that though every experiment did not see an increase in RLO, there is a "significant tendency" for species in two-species mixtures to have a 13% increase in biomass production over pure stands of species (Jolliffe, 1997). This paper is significant for use in this thesis due to the use of companion planting.

Basic concepts regarding companion planting and the effect that different plants have on the soil was provided by the National Sustainable Agriculture Information Service. The information included explanations of the various mechanisms and reasons that vegetables are companions (Kuepper, 2001). Nitrogen fixations, space requirements above and below ground, and attraction or repulsion of insects are a few of the many issues listed in the document. This gives reasonable justification for the use of companion (and antagonist) planting as one of the features of the garden layout.

Within the Garden Layout module is a basic listing of vegetables and their characteristics that is used to create a garden design. A spreadsheet named VegetableDatabase.xls is available for the user and contains more information including harvesting times and plant-specific growing tips from Beck (2008) for use with the graphical layout after it has been generated. A main feature of the layout design is the pairing of companions and using the antagonist aspects of the vegetables to determine the relative location of vegetables to areas in the garden. The sources of information for this compilation of companions and antagonists, for the many vegetables, herbs, and edible flowers, were lists from Jeavons (1996), Riotte (1998), Bird (1990), and Beck (2008). These lists were combined to formulate the inventory of companions and antagonists that appear within the Garden Layout database in the developed prototype decision support system.

The preceding literature review has exhibited that considerable information exists on the subject of urban food production and is contained within a wide range of disciplines. The existence of garden planning software has been presented but these software packages do not fully address the companion/antagonist relationship between plants, nor do they allow for an integrated shadow analysis to assist the user in determining the location of a garden plot. Therefore, it may be concluded that the need for a decision support system with the aforementioned capabilities is substantiated for those with an interest in urban food production. The development of such a decision support system is outlined in the following chapter.

Chapter 3 DSS Development

3.1 Introduction

In developing this prototype decision support system the focus has been to provide a tool for the user that will generate an urban garden plan that has employed user-provided inputs. This tool will require basic knowledge of microcomputers and straightforward instructions are provided to assist the user with the use of all background software. This DSS was also designed with Canadian growing conditions in mind, and as such, only edible vegetables, herbs, and flowers that can be grown within Canadian climates are included. Other key development requisites are for the tool to be free to the public and in an interface format that is easily understood.

The two tools embedded within this DSS are the Shadow Analysis and the Garden Layout sections. These two segments are the basis for the DSS and can be used together in order to model the sun conditions which may influence garden placement and layout, or the Garden Layout segment can be used alone for users who are familiar with their land parcel's sun conditions. The following chapter will outline the details of each of these segments and refer to User Instructions that will sum up the key functional points for the intended user.

The TotalPackage.rb file contains many Ruby scripts which collectively make up the Shadow Analysis and Garden Layout modules of this decision support system. Ruby is an object-oriented programming that is, for the most part, associated with web development. SketchUp's application programmer's interface (API) uses Ruby to communicate with SketchUp; Ruby scripts are the medium of communication and have the file extensions '.rb'. In order to use this DSS, the user is required to download the free versions of Google SketchUp (Google, 2010a), Google Earth (Google, 2010b), and GIMP (The GIMP Team, 2009). The user should make note of the file path of at least the

location of Google SketchUp in order to be able to deposit the TotalPackage.rb plugin file in the appropriate folder. After the download of Google SketchUp is complete, the user must download the plugin package and deposit it in the Plugins folder. The Plugins folder resides within the Google SketchUp 7 folder (.../Google/Google Sketchup 7/Plugins). Once the plugin has been saved, it will show up in the Plugins menu, which resides in the main toolbar, when Sketchup is loaded.

3.2 Conceptual Model

This DSS is intended to assist a homeowner to plan to grow more food more effectively on their land parcel. By providing tools to characterize sunlight conditions and a structured methodology in which to plan the planting of vegetables, a user will have the opportunity to plan for their land parcel's specific characteristics. Consideration is given to parcels that receive partial sun and/or light shade due to trees or parcel orientation. Users who enjoy decorative gardens (for front yard gardens, perhaps) or who do not have room or time to grow seedlings indoors are also considered.

The planting layouts given are a suggestion but do abide by companion garden and antagonist rules so the user is free to alter relative sizes of areas, amounts of plants per area, and the plant spacing/row configurations. The following flow chart (Figure 3.1) gives the potential user a visual representation of the inputs, outputs, and operation of the DSS.

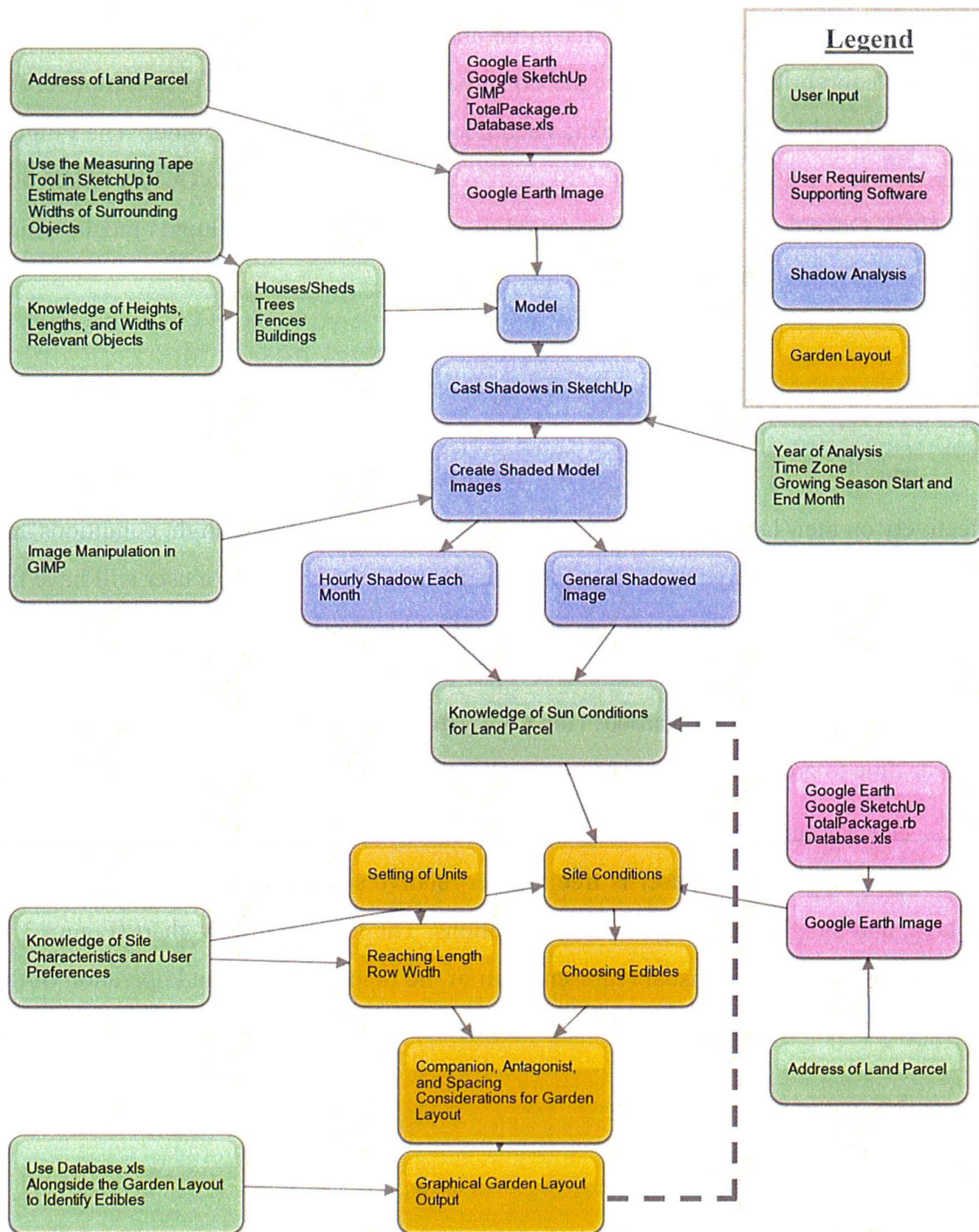


Figure 3.1 Conceptual Flow Chart of Inputs and Outputs of the Developed DSS

3.3 Shadow Analysis Documentation

The motivation behind incorporating a shadow analysis into this decision support system is essentially to give the user a sense of how sunlight will fall on the land parcel in question throughout the growing season. This SketchUp plugin is to be used along with Google Earth and GIMP 2.6, an open-source image manipulation software. The basic format, to be discussed in more detail, is to use Google SketchUp to import an image of the yard inclusive of shadow-casting objects from Google Earth, run the shadow analysis plugin in Google SketchUp, and manipulate these images to make a single image of the layered shadows. These steps will give the user a representation of the shadows during peak sunlight hours in the growing season which will allow the user to make a more informed decision on where to locate a garden on their property.

3.3.1 Using Shadow Analysis

The user (she) is encouraged to get familiar with both pieces of software by clicking on various buttons and reading the user manuals before attempting a Shadow Analysis, to gain an initial understanding of the software. To begin the Shadow Analysis, the user will open both Google Earth and Google SketchUp (SketchUp). She will open Google Earth and zoom to the appropriate location; the user should include everything (e.g. buildings, trees, fences, etc.) nearby that will contribute to shade in the area of question (see Figure 3.2). The entire image that is shown in Google Earth will appear in SketchUp (facing in the appropriate cardinal direction); therefore it is important that the user include only the area that is relevant to this analysis.

Once the viewing area (which will be the image exported to SketchUp) includes all elements that may create shadows on the relevant area, the user opens the SketchUp window and imports the image into SketchUp. There are two ways to retrieve the view in Google Earth and import it into SketchUp. The first is to find the Google Earth icon that has a yellow arrow pointing down on SketchUp's toolbar. Mousing over the icon will provoke a textbox including the phrase 'Get Current View'. Click this icon to import the Google Earth image into SketchUp. The second way to import the Google Earth image is

from the Tools menu, then the Google Earth menu, within which appears 'Get Current View'. Click this option to import the image into SketchUp.

A 2-d image will appear on the x-y plane ($z=0$) in SketchUp. This image is to scale and is oriented in the correct direction to facilitate the Shadow Analysis. This image will act as the template on which the user creates a 3-D model of the area. The user is required to have measurements or estimates of dimensions of shadow-casting fences, sheds, houses, apartment buildings, trees, etc., to be able to create an approximate model, though the Tape Measure tool can be used for determining some dimensions that can be seen within SketchUp.

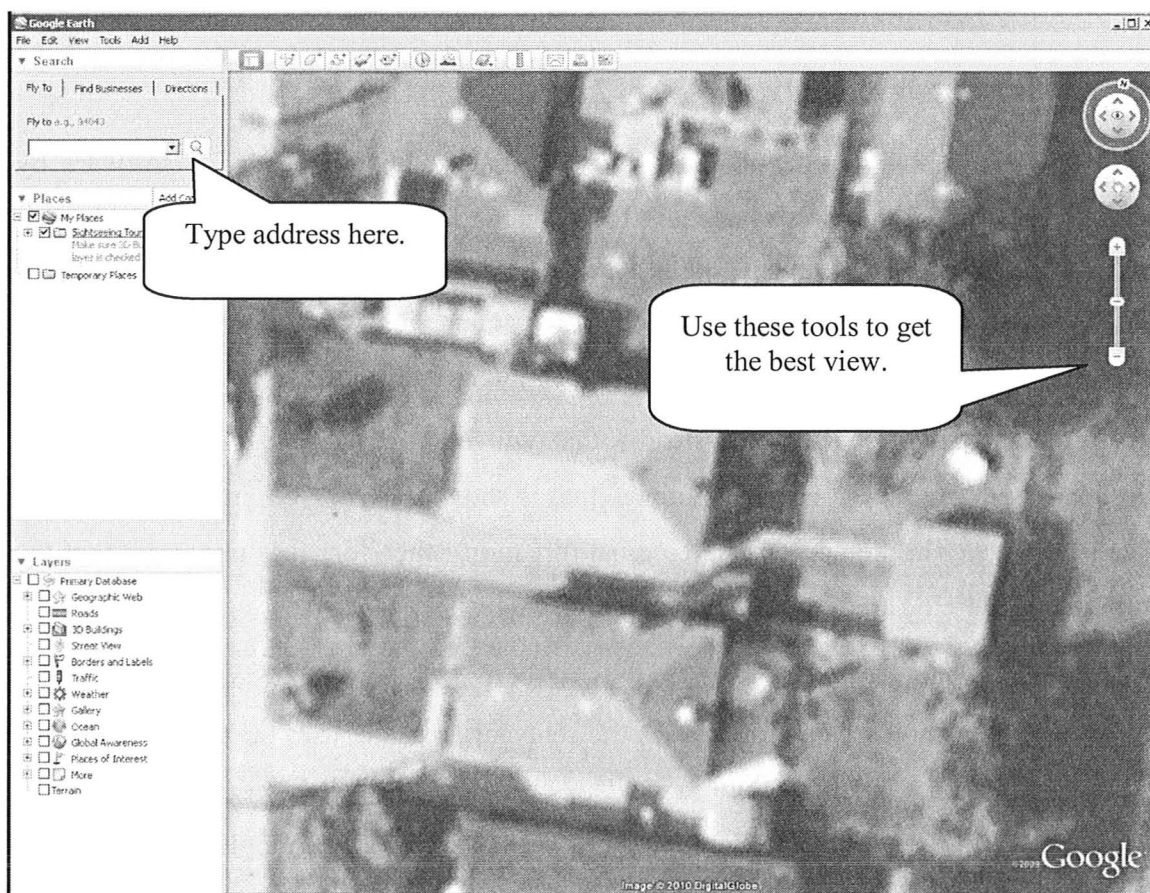


Figure 3.2 Google Earth Zoomed to Include Relevant Features

Some components are available to the user for modelling; these include simplified versions of a fence, house, warehouse/apartment building, coniferous tree, deciduous tree and a hedge. They are contained within the Shadow Analysis menu within the Plugins menu. All components will appear centered at the origin surrounded by a blue-edged box. This box will allow the user to move the component to the preferred location and perform rotations in the planes of the faces of the box, if needed. It should be noted that because the components are to appear at the origin, objects that are to be moved farthest away from the origin are to be made first to discourage overlap of components. The simplified fence (named 'Draw a Fence' within the Shadow Analysis submenu) is a 3-D structure whose user-specified length and height will appear in the x-z plane ($y=0$) and is push-pulled in the y direction to the user-specified width. The fence model is constructed with four-inch wide pieces of fencing material, the default depth is set to two inches, and the spacing between each piece is one inch.

Each of these elements are made into components, which effectively means that each component that is made cannot be dimensionally altered (the user can, however, right click the component and click 'explode' within the menu to turn the component into individual edges and faces) but can be moved and rotated, both with the Move tool. This is convenient for users that are confident in their dimensions because the components cannot be stretched in any direction. Components are also advantageous because instead of having to select all edges (potentially missing one or more edge), when the Move tool is selected the user can move the entire object as one entity without having to use the Select tool. Due to the basic shape of each of the components, if the user requires more complicated shapes, this group of components can be placed together to accomplish the task. If a more accurate rendering is wanted, the user can employ SketchUp's tools to make a model to their own detail and specifications.

These components are to be placed over the corresponding areas on the Google Earth image. When the Move tool is selected and moused over the component, a blue-edged box will encapsulate the object and red cross-hairs will appear on the faces of the box. Using these components, the user will be able to make an approximate 3-d model of

their parcel of land. This will serve to cast shadows at hourly snapshots throughout the growing season.

These images are saved as JPEGs and will be opened in GIMP as layers. By multiplying the images together and adjusting their opacity, the user will have an estimation of the sunlight falling onto their property.

3.3.2 Code Description: Shadow Analysis

Within `TotalPackage.rb` are seven Ruby methods that pertain to the Shadow Analysis portion of this DSS. The first six are: `draw_afence`, `draw_ahouse`, `draw_anapt` (a simplistic apartment building or warehouse-style building), `draw_ctree` (coniferous), `draw_dtree` (deciduous), and `draw_ashrub`. These methods produce components made from simple shapes using the user's inputted dimensions (see Figure 3.3). The `draw_afence` method produces 2 inch by 4 inch rectangular prisms whose height is inputted by the user and they repeat (separated by 1 inch) for the length inputted by the user as well. The `draw_house` produces a triangular prism atop a rectangular prism, `draw_anapt` and `draw_ashrub` produce rectangular prisms, and `draw_dtree` and `draw_ctree` produce 2-d objects: circle and rectangle, and trapezoid and rectangle, respectively. Because these two components are 2-d, their behaviour is changed to always face the camera (the user will always see the face of the object) and to always face the sun. This essentially gives the objects the property of being cylindrically symmetric without having to produce the actual 3-d object. At the end of each method, the object(s) is turned into a component, the view is zoomed to the extents of the object and that component will be the only thing selected upon creation.

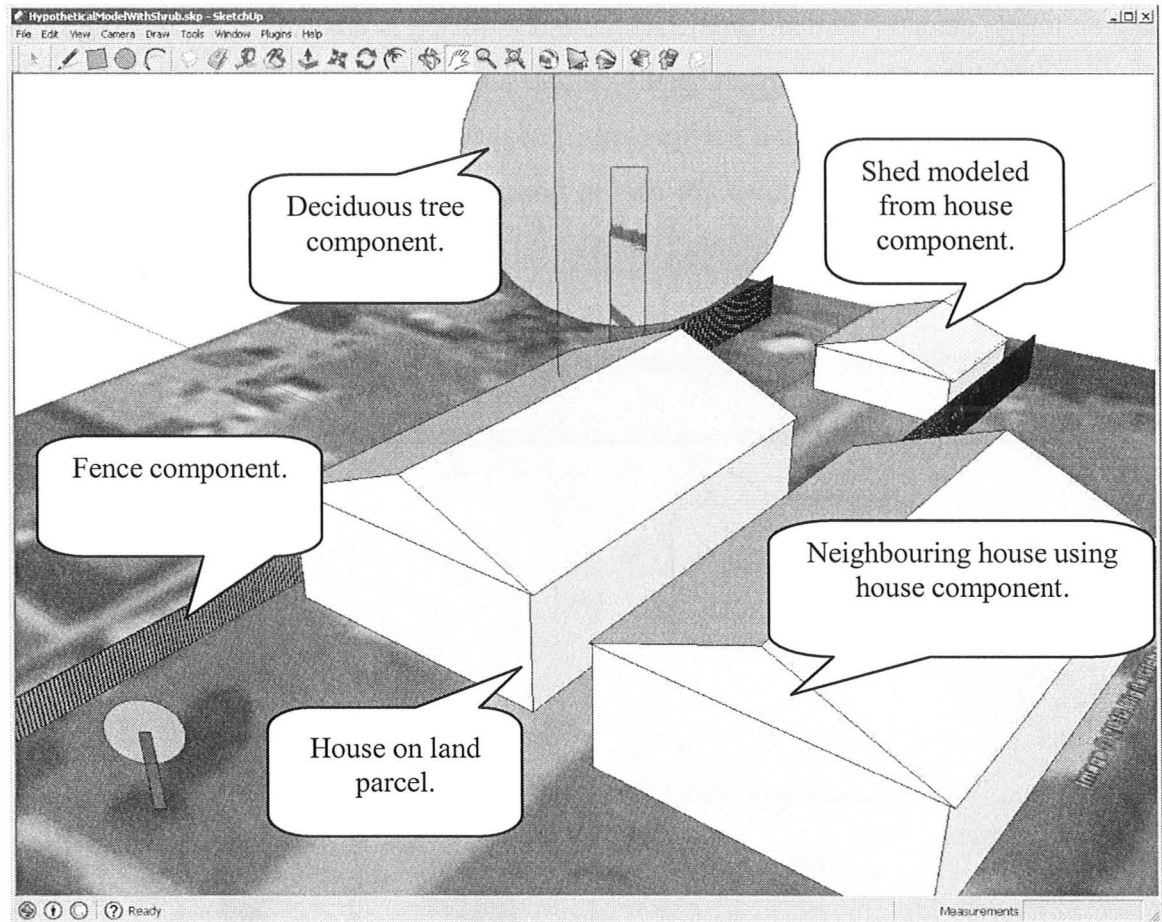


Figure 3.3 Example Model to Show How Components are Used

The last Ruby method that pertains to the Shadow Analysis module of the DSS is shadowanalysis. This method produces the images of shadows that the user will manipulate in GIMP. This method starts off by taking in the selected Google Earth image and changing the view to Camera>Standard Views>Top and zooming to the extents of the model. A series of input boxes retrieve the garden name, the year, start and end months, day, time zone, and folder name in C:\ from the user. The image is saved to the specified folder (if nothing is entered it is saved directly to C:\) as [GardenName]BackgroundMap.jpg. Each image has its own page or scene within the SketchUp window (shown in Figure 3.4) and the title of the scene is the same as the file scene's image filename. The Google Earth image that has been selected is then deleted

and an image named [GardenName]BackgroundBlank.jpg is created. Then, starting from the hour of sunrise plus 2 hours and ending at the sunset hour minus two hours (from Bell (2000) it has been inferred that the spectral irradiance is highest in the four hours above and below solar noon), shadows are cast in hourly intervals. Each scene and image is named [GardenName][Year][Month][Day][Hour] and images are saved to the user-specified location.

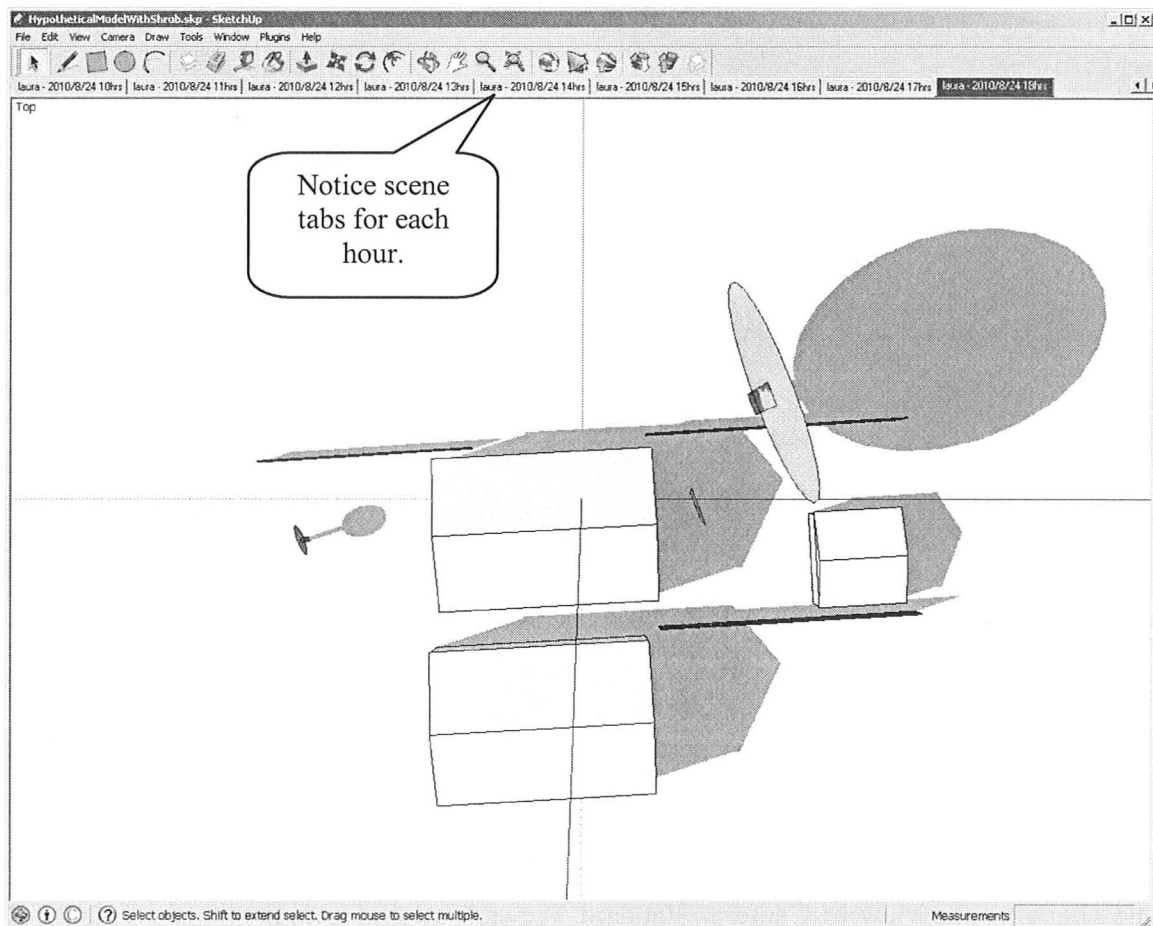


Figure 3.4 Screen Shot to Show Shaded Model in SketchUp

It is at this point that the user can manipulate these images to gain more knowledge about the sun conditions of the parcel of land. This will help the user when it comes time for her to apply the Garden Layout portion of the DSS to the same plot of land.

3.4 Garden Layout Documentation

The Ruby method, `gardenlayout`, residing in `TotalPackage.rb`, is the only piece of code that is needed to complete the Garden Layout module. Many tasks are accomplished within this method, now amalgamated to form one script with input and output flows that are appropriate for the general user. The major tasks included in this script are:

- retrieving the user-drawn/input rectangular garden plot (with user-defined size, location and orientation),
- retrieving user preferences with regard to unit convention, garden-path size and maximum reaching ability,
- retrieving a user-selected list of vegetables, which has been modified to suit the user-defined site conditions,
- providing a graphical representation of the user's garden with assigned colours and spacing organized into rows for ease of use; vegetables may be paired and are done so according to companion planting information and the layout is governed by antagonistic relationships between the vegetables.

The following section will document the details of the `gardenlayout` method, and to a lesser extent, give insight into the DSS's instructions which reside in the appendix. This method is to be used in Google SketchUp and can be accessed through SketchUp's Plugin's folder (where the user will save it to after downloading it).

3.4.1 Using Garden Layout

This plugin is to be used in one of two ways: after the Shadow Analysis module using the parcel of interest or after the Google Earth image of the parcel is imported into SketchUp (foregoing the Shadow Analysis module). In any case, SketchUp needs to be open if the user intends to perform the Garden Layout module.

In order to use the Garden Layout module, a rectangle, which is representative of the size of the intended garden, needs to be drawn on the x-y plane ($z=0$). The face (i.e.

the surface of the rectangle) and two specific edges need to be selected. If the user has completed the Shadow Analysis module, she should refer to the layered .jpg image she created in GIMP to choose the appropriate garden location(s).

The task of drawing this garden representation can be done by choosing the rectangle tool, left-clicking the drawing area, and, keeping the left button depressed, moving the mouse diagonally until suitable garden dimensions are achieved. The vertical and horizontal metric dimensions of the rectangle are visible in the bottom right-hand corner of the screen.

Many tools within SketchUp can be used to orient the rectangle in the correct way. The Move and Rotate tools will be of most use while the Zoom Extents, Measuring Tape, Pan, and Camera functions will be used sparingly. The end result is to have a rectangle (with corners of 90 degrees) of correct proportions and lying on the ground plane ($z=0$).

Once the user is satisfied with the location and orientation of the garden, she selects the face, and depressing the Ctrl button, selects the connected edges representing the foot of the garden (where the user will access the paths) and the left edge (taken from the perspective of the foot of the garden) with the Selection tool. Once these two edges and the face are selected, the user will select Plugins>Garden Tools>Garden Layout.

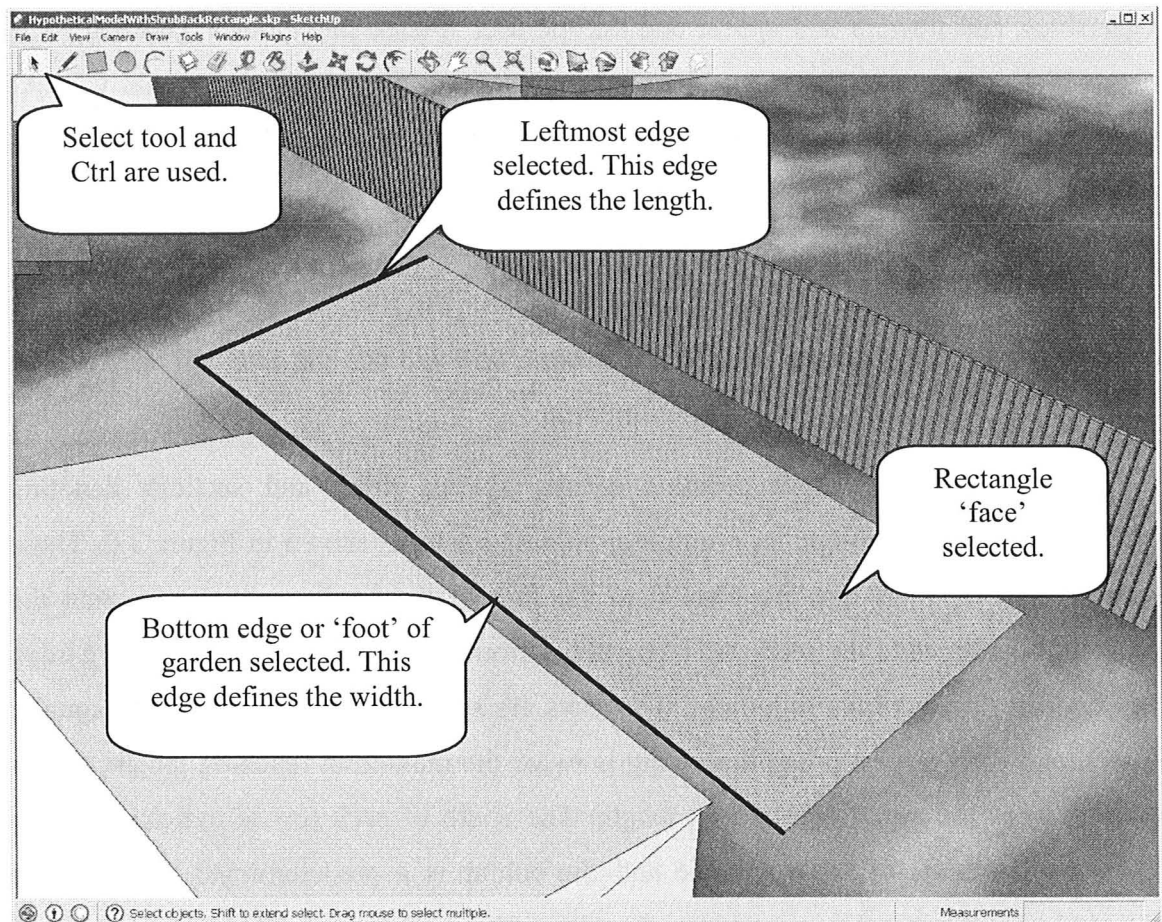


Figure 3.5 Representative Rectangle with Edges and Face Selected in SketchUp

An input box will appear with a drop-down menu which asks for preference of units: imperial (inches) or metric (centimetres). Next, she is asked to input the maximum reaching length, the greatest comfortable distance at which the user is willing to work (for weeding, picking fruit, hoeing, etc.), and path width, where the user should take into account the methods and implements in use while traversing the paths (for instance, wheelbarrows, body placement, etc.). Then, she is asked to reveal her preferences and site conditions corresponding to decoration (preference for attractive plants), sun, growing cycle length and sowing; this will alter the vegetable list to include only vegetables that adhere to the corresponding inputs. And finally, she is asked for her preference of vegetables to grow. An input box containing a drop-down menu of various herbs,

vegetables, and flowers will appear and ask the user "Please pick your vegetables". The user will choose one of the vegetables. A subsequent dialog box asking "would you like to add more vegetables to your list?" will appear (with the list of selected vegetables) to ask if she would like to pick more. This sequence will repeat until "No" is selected in the dialog box. One last dialog box will appear with the amount of 'gardenable' area (garden area minus path area) in the user's earlier chosen units. If "Cancel" is selected in any dialog box at any point in the method, the process will end and the user will have to start over again if she wishes to complete the module.

The graphical output contains equally spaced paths and sections denoting different vegetable pairings; an example graphical output is shown in Figure 3.6. These sections contain rows that are parallel to the foot of the garden, which represent the vegetables assigned to the section. The rows run from the foot to the length of the garden less two maximum reaching lengths. The rows are separated such that they are equally spaced and their largest possible spacing is twice the maximum reaching length, this is referred to as the modified reaching length. The width of each row is indicative of the space requirements of the vegetable and the colour is a predetermined unique color assigned to that particular vegetable so that the difference between vegetables within and between sections is immediately apparent. The colour and corresponding vegetable information is contained within a Microsoft Excel spreadsheet, VegetableDatabase.xls. This will allow the user to determine the vegetable layout and other relevant information such as plant height, harvesting requirements, gardening tips, etc.

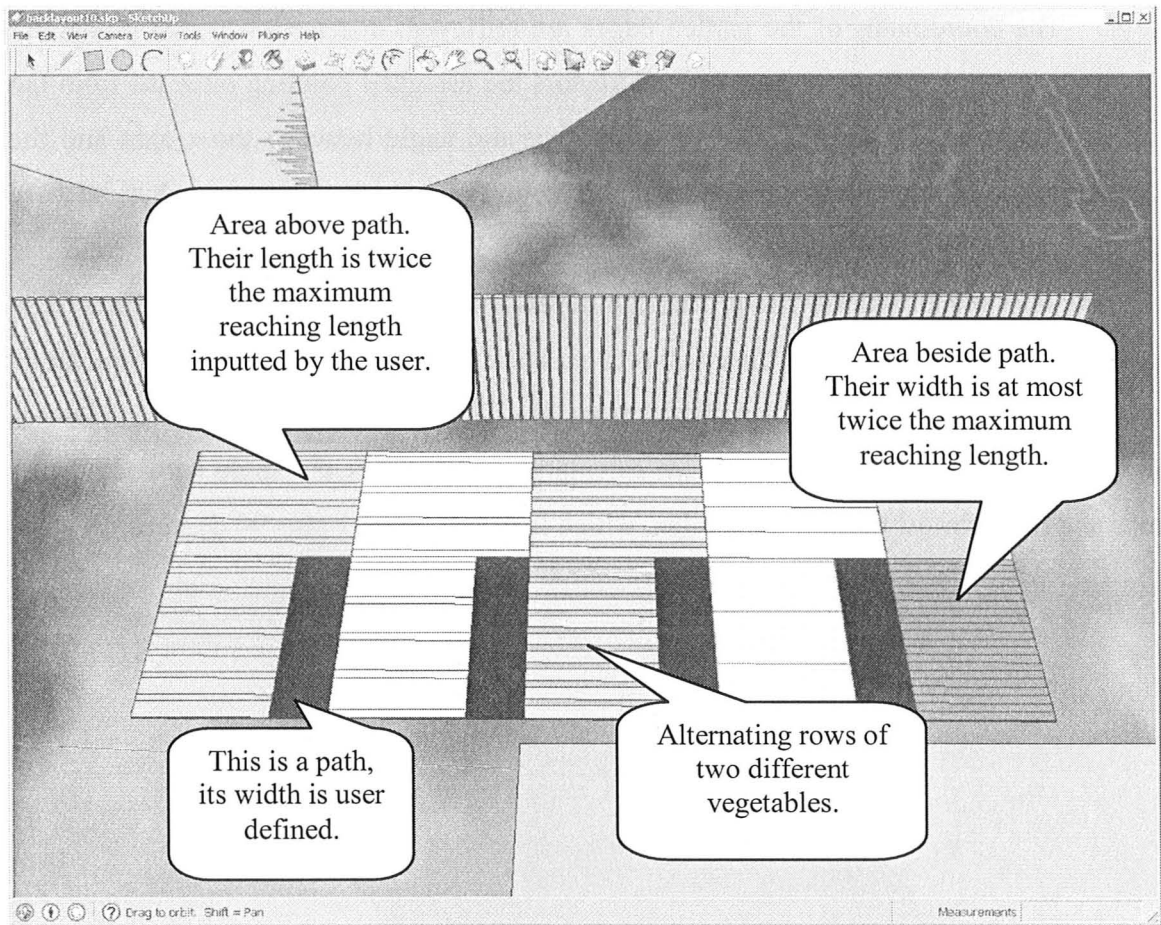


Figure 3.6 Screenshot of the Garden Layout Showing How/Where Quantities are Used

3.4.2 Code Description: Garden Layout

TotalPackage.rb contains one method that pertains to the Garden Layout module: `gardenlayout`. This method retrieves the coordinates and orientation of the user-selected rectangular face and two edges. In order to ensure the user has followed the proper steps regarding drawing the rectangle and selecting the appropriate items, if-loops count the number and check the types of items selected before any operations are carried out. These if-loops make use of the variable `selout`, which is an array of the selected items, to ensure that the number of items is three and that the selected entities are, in fact, edges and a face. If these requirements are not met, a dialog box containing a descriptive error message will appear and the process will cease.

The coordinates of the garden edges are retrieved: distances between points are gathered using the distance function and vectors are assigned pointing outward from the vertex of the edges. **Angle1** (see Figure 3.7) is the angle between the x-axis and the bottom-edge of the garden measured in a counter-clockwise direction, but because SketchUp only returns angles between 0 and π (i.e. the angle is measured from the clockwise or counter-clockwise orientation depending on whether the angle is above 180 degrees), the cross product of the positive x-axis and **angle1** is taken and depending on its orientation, the angle may be subtracted from 2π . This allows for the proper orientation in the graphical layout and subsequent calculations. **Angle2** is the angle between the x-axis and the leftmost edge of the garden, which is simply **angle1** plus 90 degrees.

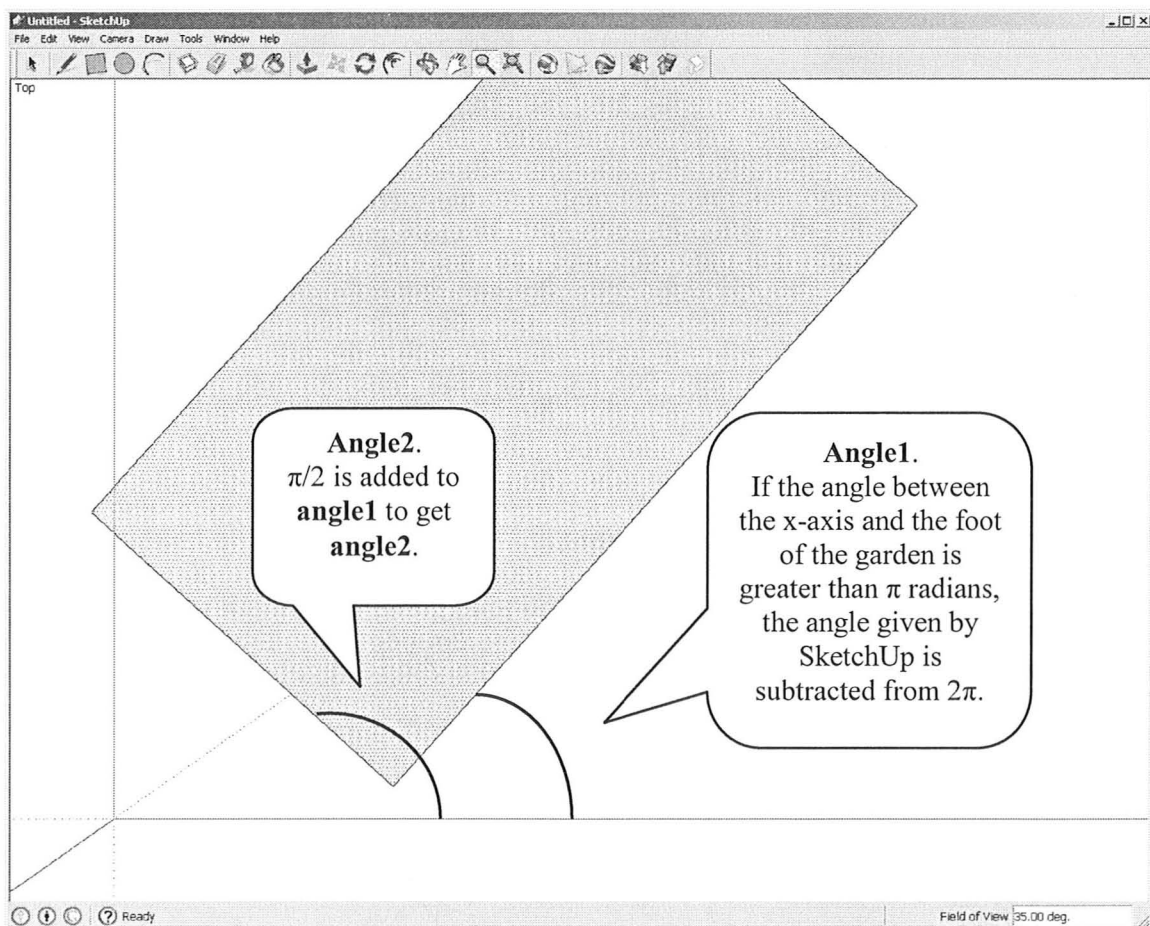


Figure 3.7 Diagram showing the angles **angle1** and **angle2**.

To be able to configure the garden space to the user's specifications and preferences, the script requires information from the user. Input boxes are used to gather information externally and require a user prompt, which holds the questions or prompts to be displayed in the dialog box; default values, which appear in the input field and can be empty strings, strings, or numbers; and the `UI.inputbox` command which includes these two items and may contain a list item, if a drop-down menu is preferred. The first input box asks for a unit preference; this information is contained within a drop-down list containing imperial (inches, default) or metric (centimetres). The `UI.inputbox` command is an array where the number of inputs is the length of the array. Following this input box, another box appears and, using the previous unit information, asks for maximum comfortable reaching distance and preferred path width.

Using information from a variety of resources, a list containing common vegetables suitable for Canadian climates as well as relevant gardening information such as companions, antagonists, spacing, etc is contained within the `gardenlayout` method. Within the 'vegetable database', each vegetable is organized into what Ruby calls a hash. Hashes have a key that is assigned a value. An array is analogous to a hash: an array has an implicit index corresponding to each value, whereas a hash's 'key' is explicitly defined by the programmer and can be a number, string, array, etc. Currently, the keys in each hash correspond to: vegetable name, antagonists, companions, spacing, sowing conditions, sun conditions, decoration, soil conditions, growing length and assigned colour.

One of the most important tasks in this decision support system is choosing the vegetable layout within the garden area. The user is prompted to choose a vegetable within an input box's drop-down menu hosting all vegetables in the database. One vegetable is to be chosen at a time. And after a vegetable is picked, a cumulative list of chosen vegetables is presented and the user is asked if she wants to continue selecting vegetables: 'Yes' and 'No' buttons will allow her to continue or terminate selecting vegetables, respectively. When the user chooses the 'No' button, the corresponding while loop is broken and a vegetable list is then converted into an array of corresponding

hashes and their companion and antagonist lists are modified to include only the vegetables from the user-selected list.

This array, **vegarray**, is to be sorted in ascending number of companions to allow vegetables with fewer companions to get paired before their companions are alternatively assigned. Before this happens, however, each vegetable's spacing is compared to the length of the two types of areas in the garden. If the spacing is greater, the vegetable will not be paired. Also, within the companion pairing loop, another loop ensures the combined spacing of a pair is less than the lengths of the gardening areas. Space requirements greater than the length of an area may cause graphical problems in the output.

The pair assignment process is as follows: the first entry, the vegetable with fewest companions, is chosen from **vegarray**. This vegetable name is searched within the companion lists of each vegetable remaining in the array (still in ascending companion number). When a match is found, the name of the vegetable whose companion list contained the name of the first entry is queried and the two hashes are put into a two-element array. If there are no mutual companions in this array, its order will stay the same and no vegetables will be paired. This array is entered twice into the paired vegetable list, **pairle**, to give the same representation to paired and unpaired vegetables. If a vegetable's name is not present in any companion lists, it is paired with an empty hash (not paired with another vegetable) and subsequently stored as one element in **pairlisto**. After each vegetable is put into **pairlisto** it is deleted from **vegarray** to prevent vegetables from being paired to more than one vegetable. After each pair is made and stored, the companion lists are updated and the process continues until there are no remaining vegetables in **vegarray**. Each pair has been added to **pairlisto** twice (once in the first position and once in the last position of the array) and each unpaired vegetable has been added only once to the last position of the array. **Pairlisto**'s order is reversed and this is now defined as **pairlist**. **Pairlist** was defined as such because it will later be multiplied so that the number of array entries is equal to the number of areas in the garden; this

arrangement of **pairlist** will make it most likely that each vegetable will be represented in the garden layout.

Now that the vegetables are appropriately paired, the focus is turned to the user-drawn rectangle (the model of the garden), and how and where to place the vegetables within this area. First, the rectangular plot is sectioned into paths and gardenable areas. The coordinates of the corners of the rectangular garden plot, along with the user's preferences for path width and maximum reaching length, are used to make equally spaced paths running from the foot of the garden to two maximum-reaching-lengths from the end (see Figure 3.8). The width of the garden is defined by the foot of the garden and the leftmost side of the rectangle defines its length. The spacing between the paths is, at a maximum, the input maximum reaching length. The number of paths is the width of the rectangle minus twice the maximum reaching length, divided by twice the maximum reaching length plus the path width; if this number is exactly an integer, then the distance between paths is exactly twice the maximum reaching length. If the number of paths is non-integer, then this number is rounded up to the next integer and the path widths are subtracted from the rectangle width and divided by the number of paths plus one to calculate the distance between each path, which is twice the modified reaching length. The garden layout is such that all of its bounding edges are expected to be accessible from the perimeter of the garden. It is for this reason that the top-edge is two maximum reaching lengths from the top of the path, and the sides of the garden are two modified reaching lengths from the paths, being the same as inter-path distances.

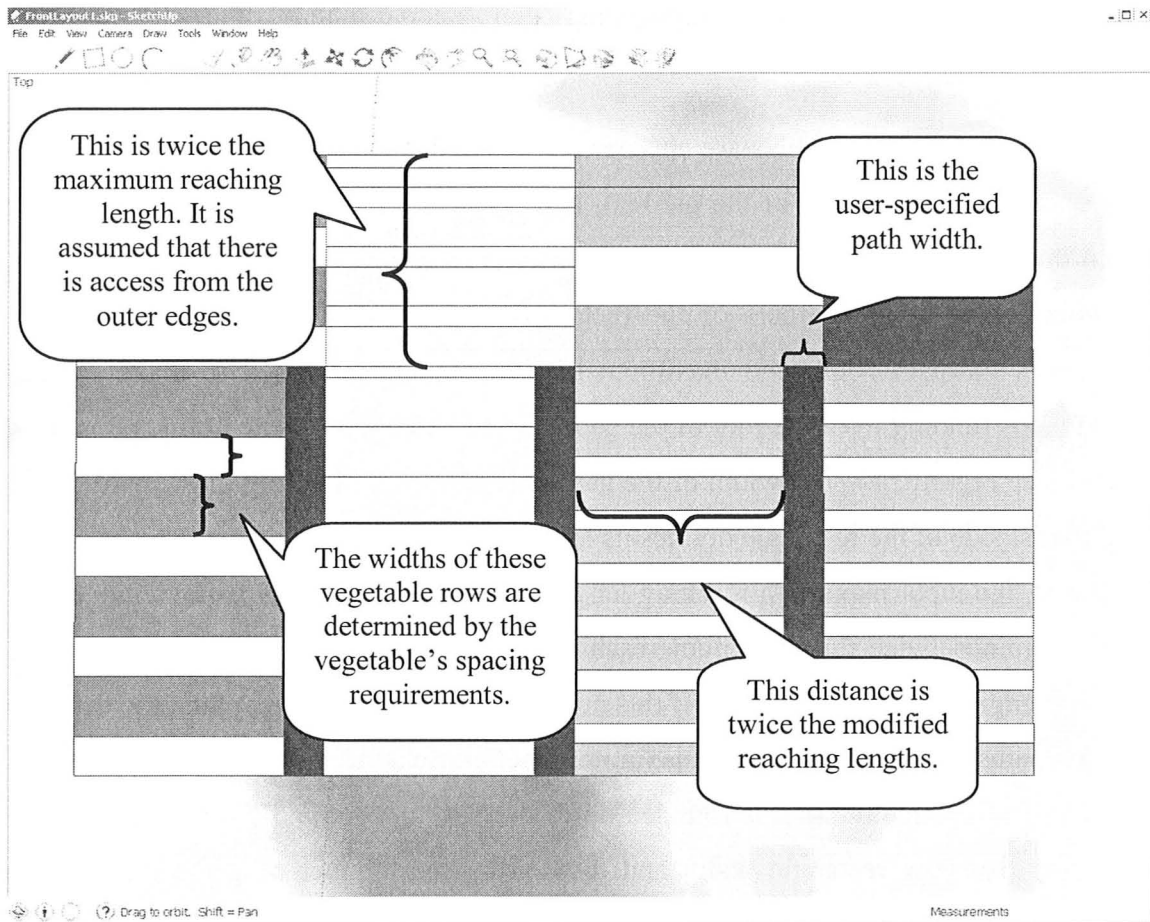


Figure 3.8 Diagram of the graphical layout showing the lengths of its elements.

The 'gardenable' part of the garden plot is the rectangle area minus the path areas and refers to the areas where food is intended to be planted and grown. The garden is split into two types of areas: areas beside the paths and areas above the paths. These areas are delineated by a line drawn across the width of the rectangle which lies on the tops of all paths combined with the extension of the right edge of the paths. Using these separations, the areas are calculated, and depending on their ratios, the areas may be split into two or more sections in order to obtain area ratios closer to one.

The coordinates of the corners of these areas within the rectangular plot are put into an array, **alls**, such that the first area in the list is the top leftmost area. The areas will be assigned vegetable pairs and are coloured in rows according to the vegetables'

assigned colours. The direction of filling is along the length in sequence toward the foot of the garden, the filling then starts from the bottom (on the other side of the first path) and fills upward toward the top. The filling continues in this serpentine pattern.

Once the list of areas (**pairlist**) is in the specified order, **pairlist** is multiplied such that the number of areas is equal to the number of vegetable pairs, this array is called **pairle**. The vegetable pairs are now assigned to the areas, in order, according to antagonist influences. The general procedure for assigning vegetable pairs to areas is as follows: a for-loop selects an area from **alls** in order of the indexing, this being the *nth* area. The coordinates of the *nth* area are compared to the coordinates of all assigned areas. The previously assigned areas reside within an array, **edge**, with each element in **edge** being a vegetable hash corresponding to the assigned areas that share an edge with the *nth* area. In order to find a suitable vegetable pairing for the particular configuration, or more directly, a vegetable pair that is not adjacent to any of its antagonists, the names of these bordering vegetables are searched within the antagonist list of each vegetable in **pairlay** until one is found to have none of these surrounding vegetables. Once this happens, that vegetable is assigned to the area and the cycle continues until the end of the area array. If all vegetables in **pairlay** are found to have a non-zero number of adjacent antagonists, then the vegetable pair with the minimum number of antagonists is selected and if there are many vegetable pairs with that same minimum number, the last one to have accumulated these antagonists is picked. This number of antagonists resides in the variable **ant**, and corresponds to a complete hash (**aunt**) where the key is the coordinate array and the value is the vegetable hash.

Due to the possibility of a garden layout having a non-zero number of areas containing adjacent antagonists, the script utilizes multiple garden 'scenarios'. These scenarios are actually various ways of organizing **pairlay**; this serves to change the order in which the vegetables are assigned to areas. Before the area-assigning procedure starts, a for-loop is initiated and **pairlay** is sorted in a different way for each of the five iterations of the for-loop. But once one of the five different scenarios yields zero adjacent antagonists in the layout, the loop is broken and this layout is chosen.

The first scenario is sorted such that the number of antagonists per vegetable pair is listed in descending order first and then sorted in alphabetical order to clump duplicate vegetables together. If it happens that the particular group of chosen vegetables has no antagonists in their modified antagonist lists, the array will be sorted in alphabetical order only; there will be only one scenario in this case. The second scenario is sorted by the number of antagonists per vegetable pair in descending order only; the third scenario is sorted by number of antagonists in ascending order and alphabetically; and the last two scenarios are randomly sorted versions of **pairlay**. Figure 3.9, Figure 3.10, Figure 3.11, Figure 3.12, and Figure 3.13 are examples of the layouts with respect to different scenarios using vegetables that have no antagonistic relationships with each other. Each of the inputs is exactly the same and the vegetables used were: carrot, celery, chard, eggplant, lettuce, spinach, and tomato. For each of the scenarios all **ant** values (**ant** is a variable that stores the number of times adjacent areas contain antagonists) are added together, if the value is greater than zero this **ant** value and corresponding garden layout (**ant**) is stored in a hash called **diffscen**. Until there is a layout whose **ant** values adds to zero, a layout will be stored for each scenario of **pairlay**. If there is no layout that gives a cumulative **ant** value of zero, the layout corresponding to the minimum **ant** value of the five scenarios is used.

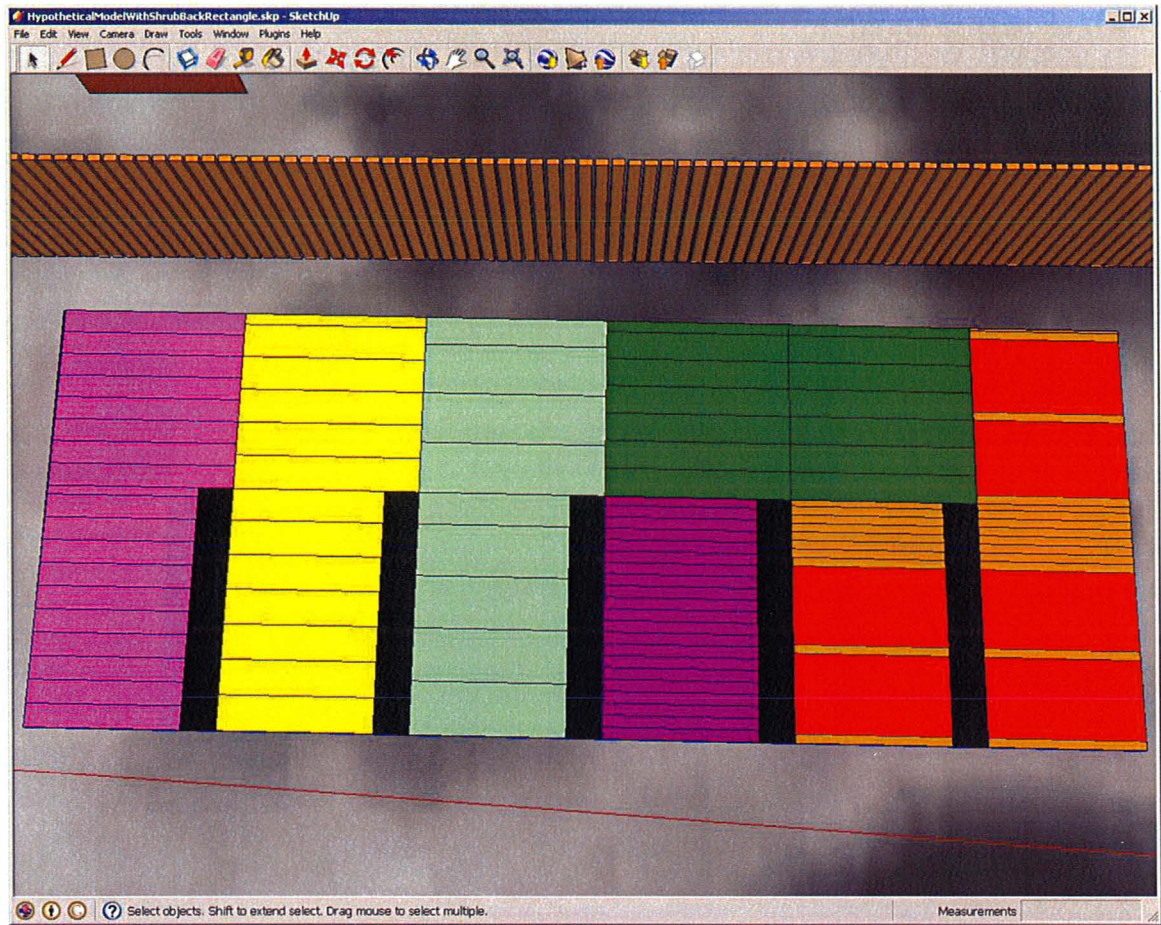


Figure 3.9 Scenario One: **pairlay** sorted in descending order of combined antagonist lists and name.

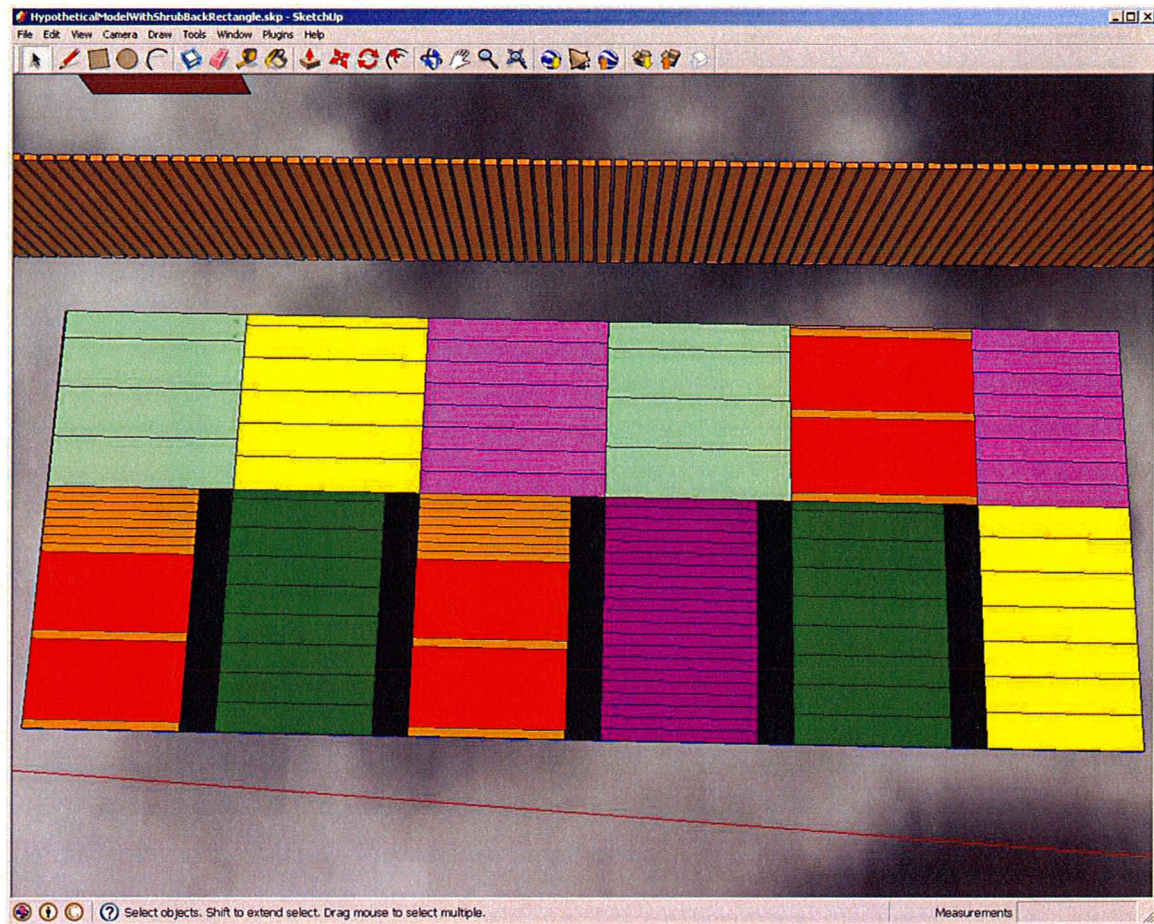


Figure 3.10 Scenario Two: **pairlay** sorted in descending order of combined antagonist lists.

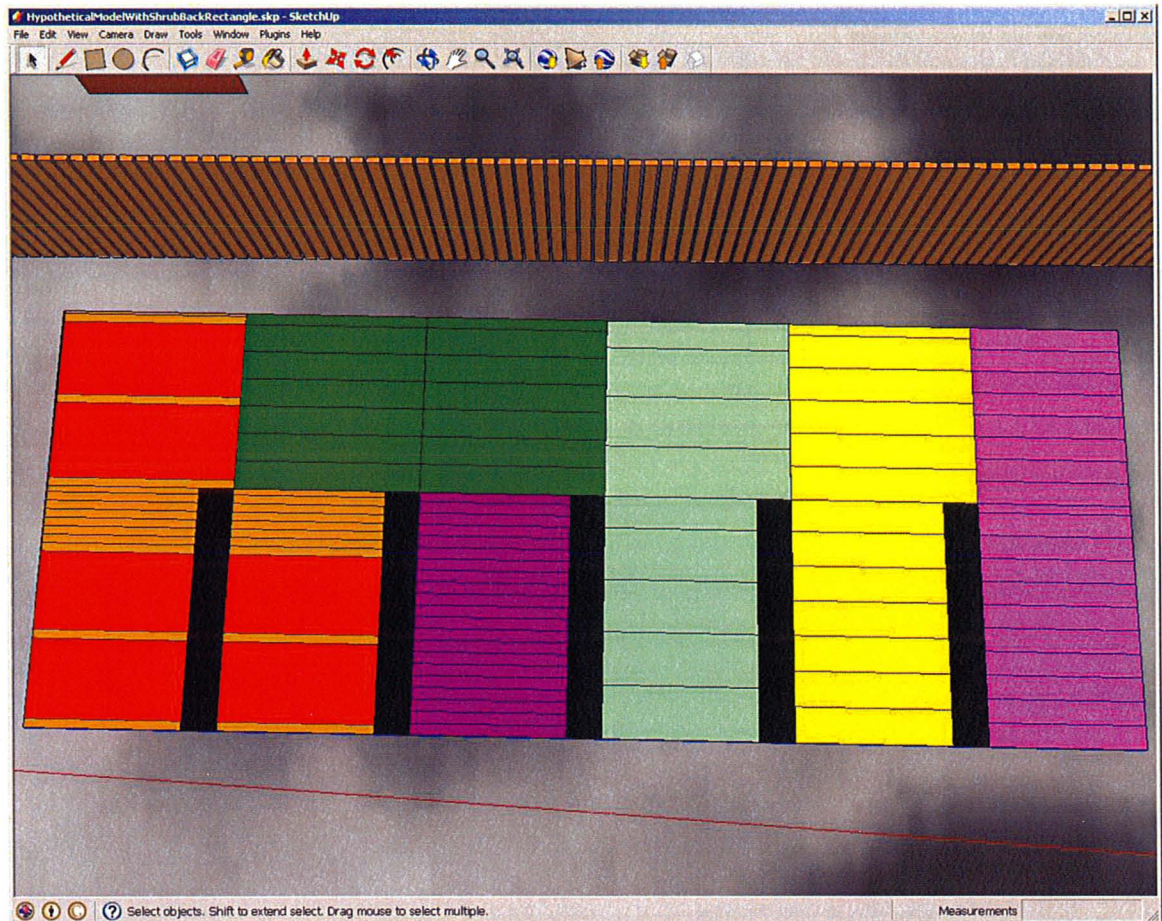


Figure 3.11 Scenario Three: **pairlay** sorted in ascending order of combined antagonist lists and name.

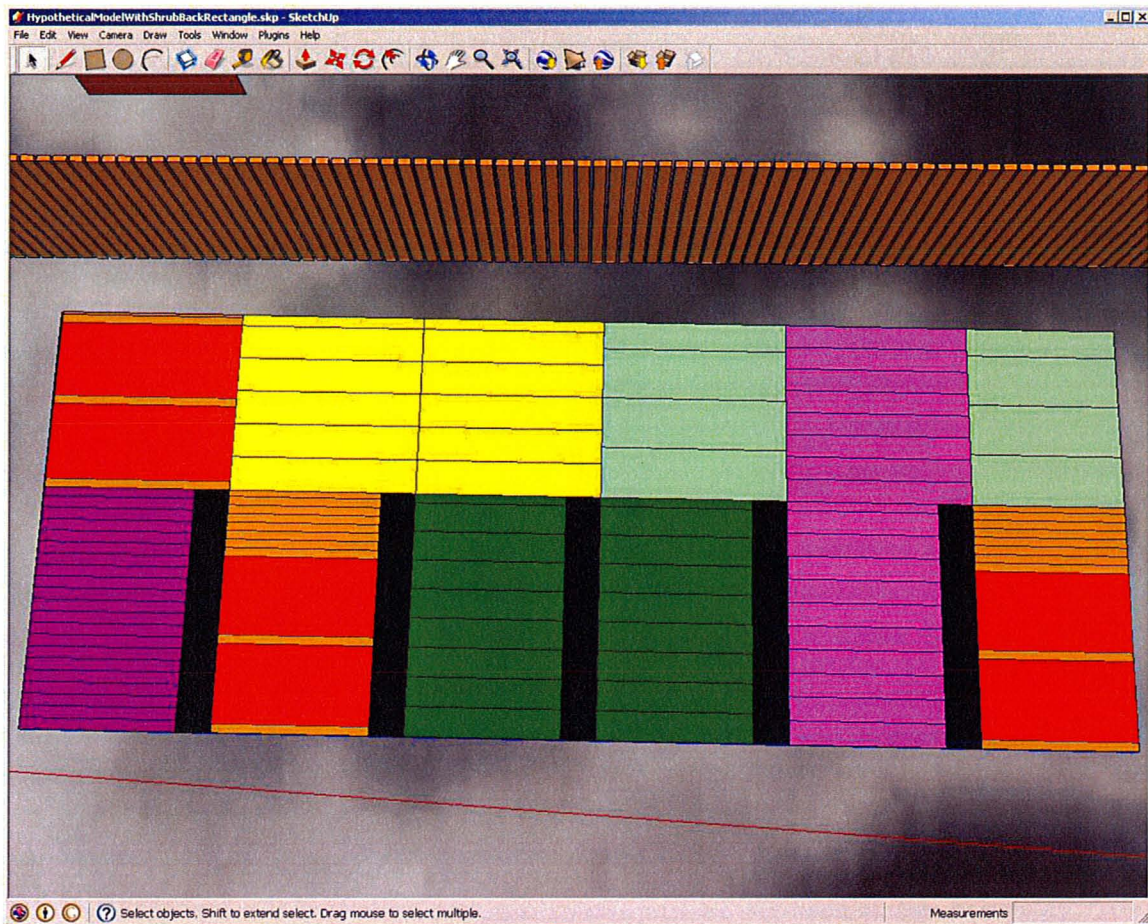


Figure 3.12 Scenario Four: **pairlay** sorted in random order.

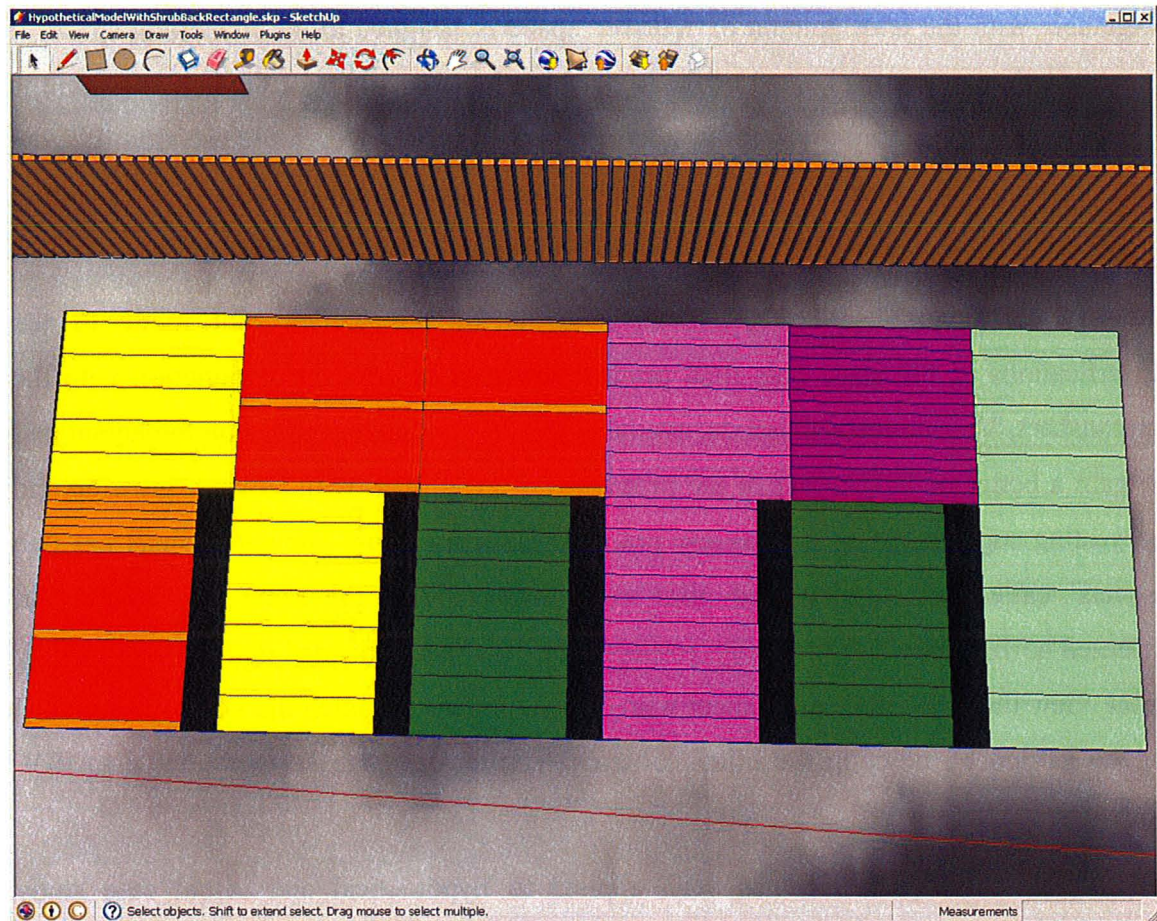


Figure 3.13 Scenario Five: **pairlay** sorted in random order a second time.

Once the garden layout is chosen, the graphical representation is created one area at a time. The process starts by calculating the number of times each vegetable (using their plant spacing) can be fit into the length of the area since each vegetable row is parallel to the foot of the garden or the width. Each row is coloured according to that assigned in the hash; the `face.back_material` command is used to assign the face a colour. The areas are filled with alternating horizontal rows of the vegetables with a row width representative of each vegetable's spacing requirements. The remaining length of the plot is obtained by subtracting the integer number of the combined spacing of the vegetables from the area length. After the integer number of vegetable spacings has been assigned, each vegetable spacing (or multiples of) is subtracted from the remaining space and the

vegetable with the least amount of 'leftover' space is assigned to the remaining space. If there is leftover area at the end of the vegetable plot, the colour of the last vegetable is filled into the remaining piece for aesthetic appeal. If neither vegetable can fit a full row into the remaining area, the last vegetable's colour will be assigned to that remaining area.

The outcome is a representative layout of the garden plot made to the specifications of the user. This final product can now be used in conjunction with the spreadsheet, *VegetableDatabase.xls*, available to identify the vegetables on the layout and to gain a better understanding of the properties of each plant with sowing, growing and harvesting suggestions as well as any practical gardening tips. The spreadsheet has two 'sheets' called *screen_view* and *print_view*; they contain the same information but oriented horizontally and vertically, respectively. It is recommended that the user modify *print_view* (and save the spreadsheet with another name) to match the vegetables picked in the garden by deleting the other columns. The entire database is available in PDF format as well.

The developed DSS has been applied to hypothetical and actual case study applications and this is documented in Chapter 4. Each actual case application includes user comments and suggestions from the DSS as well as any modifications based on user recommendations.

Chapter 4 Applications of the Developed DSS

4.1 Introduction

The purpose of this prototype DSS is to act as a tool and a resource for individuals that are interested in producing a portion of their own vegetable, herb, and/or edible flower needs in a defined land parcel. In order to ensure the ease of use and clearness of instructions of this DSS, as well as learning its limitations, hypothetical and actual case studies were carried out. The hypothetical case study consisted of a model of a representative lot in Hamilton, Ontario, where the Shadow Analysis plugin was applied and front and back yard locations for gardens were selected. The Garden Layout plugin was applied to these garden sites multiple times in order to show variation in the parameters of the plugin and to understand more about the DSS's limitations. The actual case studies consisted of three individuals using the instructions similar to the ones included in the Appendix (the instructions included in the appendix were modified using user feedback) to plan a garden using a Google Earth image of their property as a template.

The following chapter describes the input conditions of the case studies and the information pertaining to the cases are tabulated with representative images.

4.2 Hypothetical Application

4.2.1 Shadow Analysis

In this hypothetical case study a representative parcel of land was used to show the features and limitations of this DSS. The Shadow Analysis plugin served to act as a guide in characterizing sun conditions of a given land parcel. As seen in Figure 4.1, the

house and adjacent house were modeled, as well as a shed, fencing and surrounding trees. All components were made using the components within Plugins>Shade Tools.

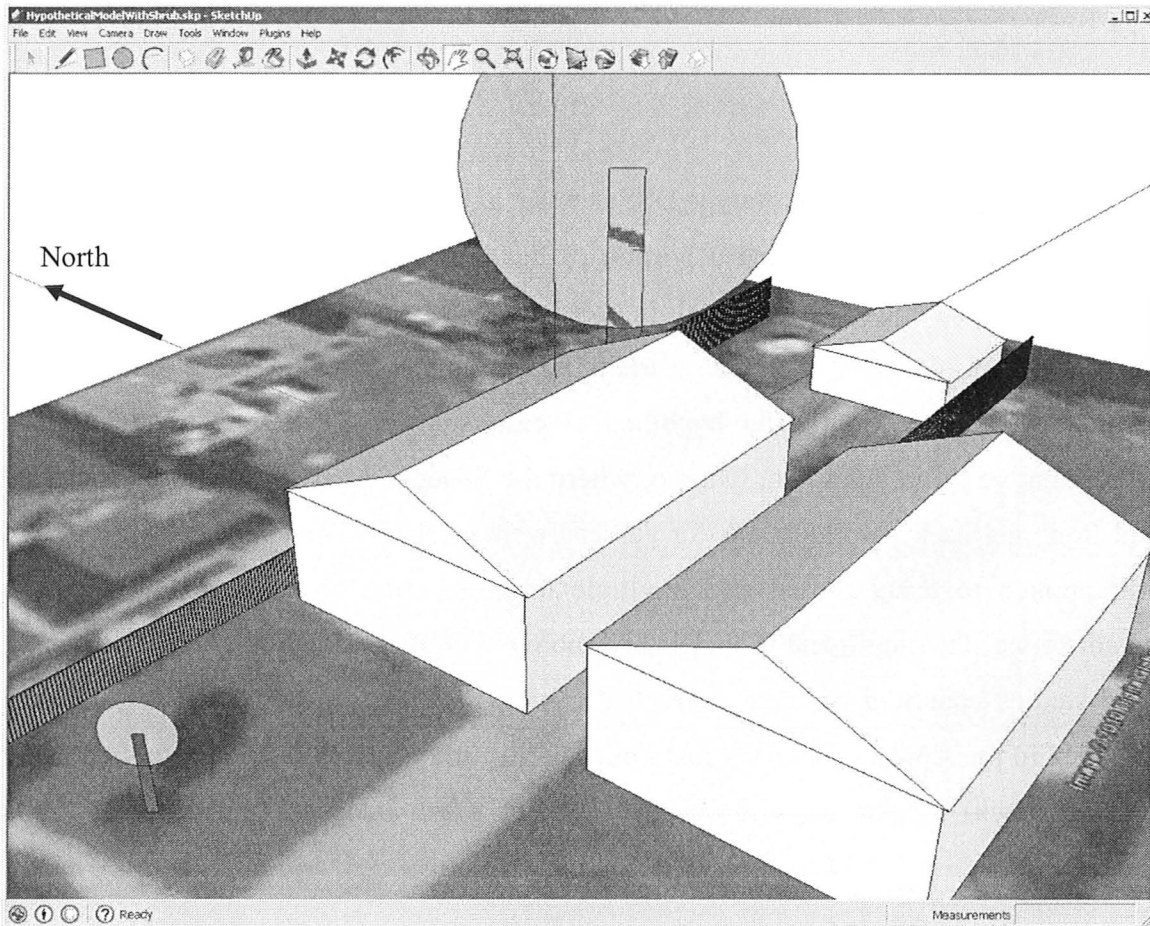


Figure 4.1 Model of Representative Parcel.

Figure 4.2 shows the combined hourly (minus two hours after sunrise and two hours before sunset) for one day in each of the months of May, June, July, and August. The bright area on the north-eastern section (full sun) of the backyard was chosen as the site for the backyard garden case study and the eastern section (full or partial sun) of the front yard is the area of interest in the front yard garden case study.

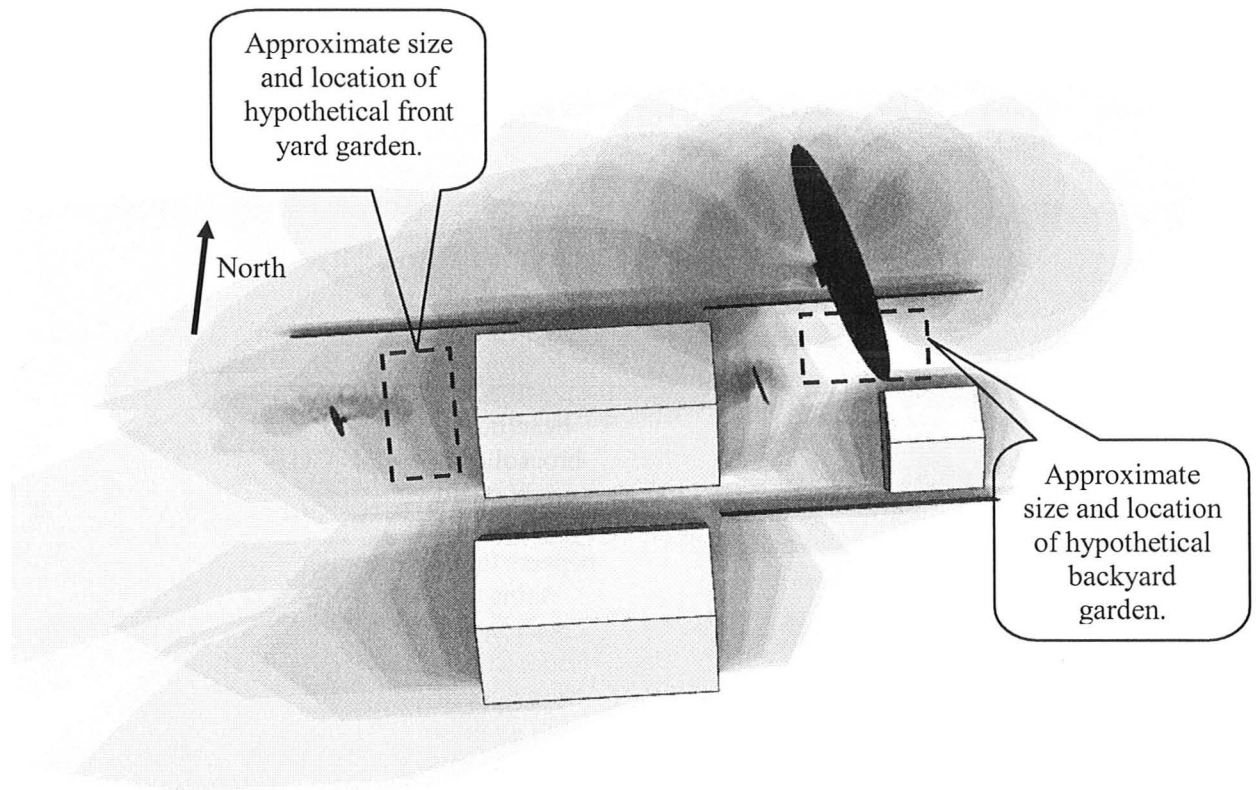


Figure 4.2 Image After Shadow Analysis (May through August, 2010).

4.2.2 Backyard Garden

Table 4.1 includes documentation of the relevant inputs (maximum reach, path width, site conditions and vegetables) of each of the backyard trials as well as the trial number, gardenable area (rounded area of the rectangle minus path area), and any relevant notes. Trials one through eleven cover ranges of maximum reaching length and path width, changing one variable at a time. From trials 12 to 21, site conditions are varied and only full sun is used in the Sun Conditions category. Some representative screenshots of the trials are shown following Table 4.1.

Table 4.1 Backyard Garden Case Study

Trial	Max. Reach (in.)	Path Width (in.)	Site Conditions (Decorative, Sun, Type , Sowing)	Vegetables	Gardenable Area (ft²)	Notes
1	32	12	All All All All	bean(bush) broccoli beet carrot chard eggplant onion peppers tomato spinach	352	
2	36	12	All All All All	bean(bush) broccoli beet carrot chard eggplant onion peppers tomato spinach	361	
3	28	12	All All All All	bean(bush) broccoli beet carrot chard eggplant onion peppers tomato spinach	348	
4	24	12	All All All All	bean(bush) broccoli beet carrot chard eggplant onion peppers tomato spinach	337	
5	18	12	All All All All	bean(bush) broccoli beet carrot chard eggplant onion peppers tomato spinach	312	
6	12	12	All All All All	bean(bush) broccoli beet carrot chard eggplant onion peppers tomato spinach	284	

7	32	14	All All All All	bean(bush) broccoli beet carrot chard eggplant onion peppers tomato spinach	346	
8	32	16	All All All All	bean(bush) broccoli beet carrot chard eggplant onion peppers tomato spinach	349	
9	32	18	All All All All	bean(bush) broccoli beet carrot chard eggplant onion peppers tomato spinach	345	
10	32	24	All All All All	bean(bush) broccoli beet carrot chard eggplant onion peppers tomato spinach	331	
11	32	30	All All All All	bean(bush) broccoli beet carrot chard eggplant onion peppers tomato spinach	317	
12	32	12	Yes All All All	bean(bush) beet carrot chard eggplant peppers	352	No antagonists present. Not all of original vegetable selection available in list.
13	32	12	Yes All All All	amaranth artichoke(globe) bean(bush) beet basil carrot chamomile eggplant peppers chard	352	No antagonists present. Not all of original vegetable selection available in list.

14	32	12	All Full All All	bean(bush) broccoli beet carrot chard eggplant onion peppers tomato spinach	352	
15	32	12	All All Annual All	bean(bush) broccoli carrot eggplant peppers tomato spinach	352	Not all of original vegetable selection available in list.
16	32	12	All All Perennial/ Biennial All	beet carrot chard eggplant onion strawberry cabbage celery fennel kale	352	Not all of original vegetable selection available in list.
17	32	12	All All All Indoors	broccoli eggplant onion peppers tomato basil Brussels-sprouts celery chives corn	352	Not all of original vegetable selection available in list.
18	32	12	All All All Outdoors	broccoli beet onion bean(bush) basil chard pea carrot spinach collards	352	Not all of original vegetable selection available in list.
19	32	12	Yes All Perennial/ Biennial All	beet carrot chard eggplant artichoke(globe) cabbage collards Jerusalem- artichoke fiddleheads leek	352	No antagonists present. Not all of original vegetable selection available in list.
20	32	12	Yes All All Outdoors	bean(bush) beet carrot chard quinoa amaranth basil bean(pole) chamomile kale	352	Not all of original vegetable selection available in list.

21	32	12	All All Annual Outdoors	bean(bush) broccoli carrot spinach pea basil cauliflower chives cucumber lettuce	352	Not all of original vegetable selection available in list.
----	----	----	----------------------------------	---	-----	---

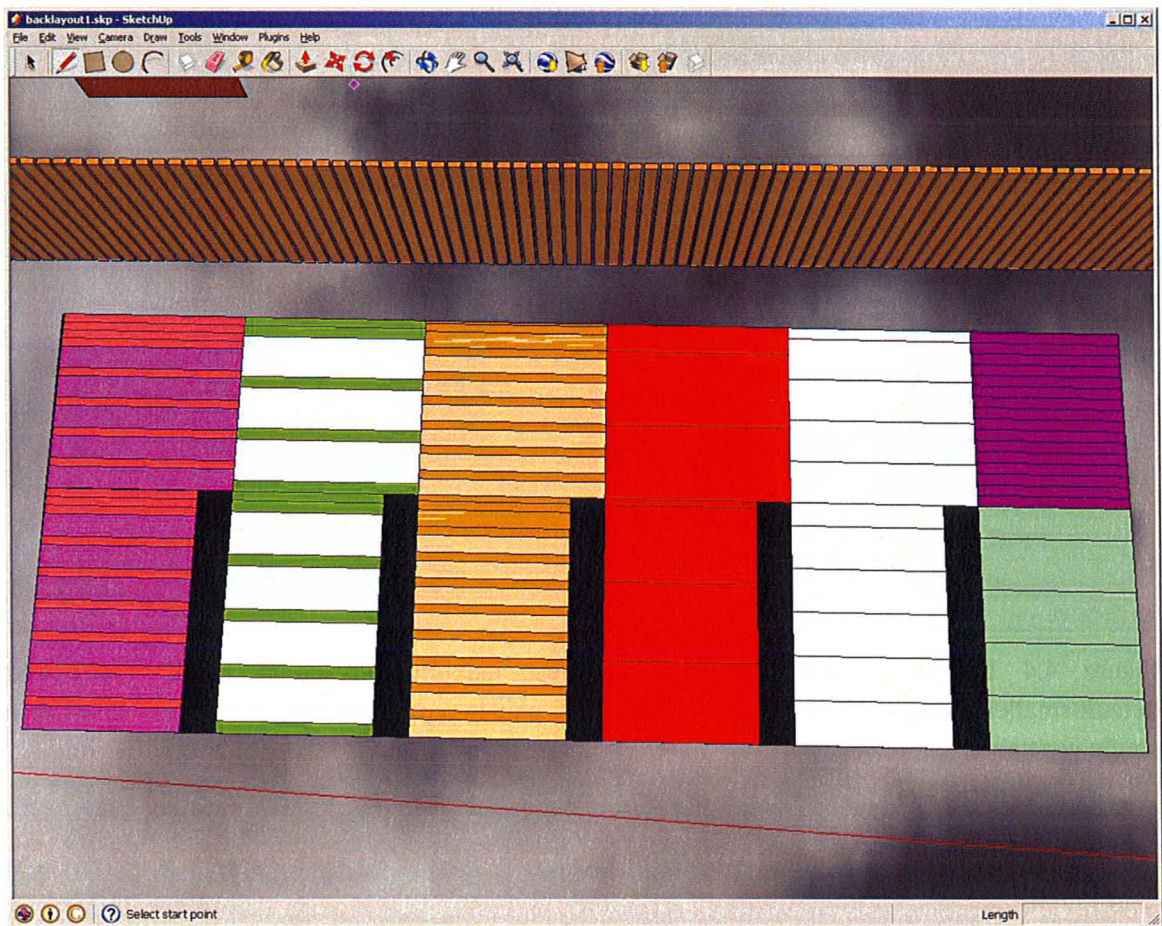


Figure 4.3 Backyard Garden: Trial 1. Default settings for reaching length and path width (32" and 12", respectively). This path width also coincides with the smallest path width.

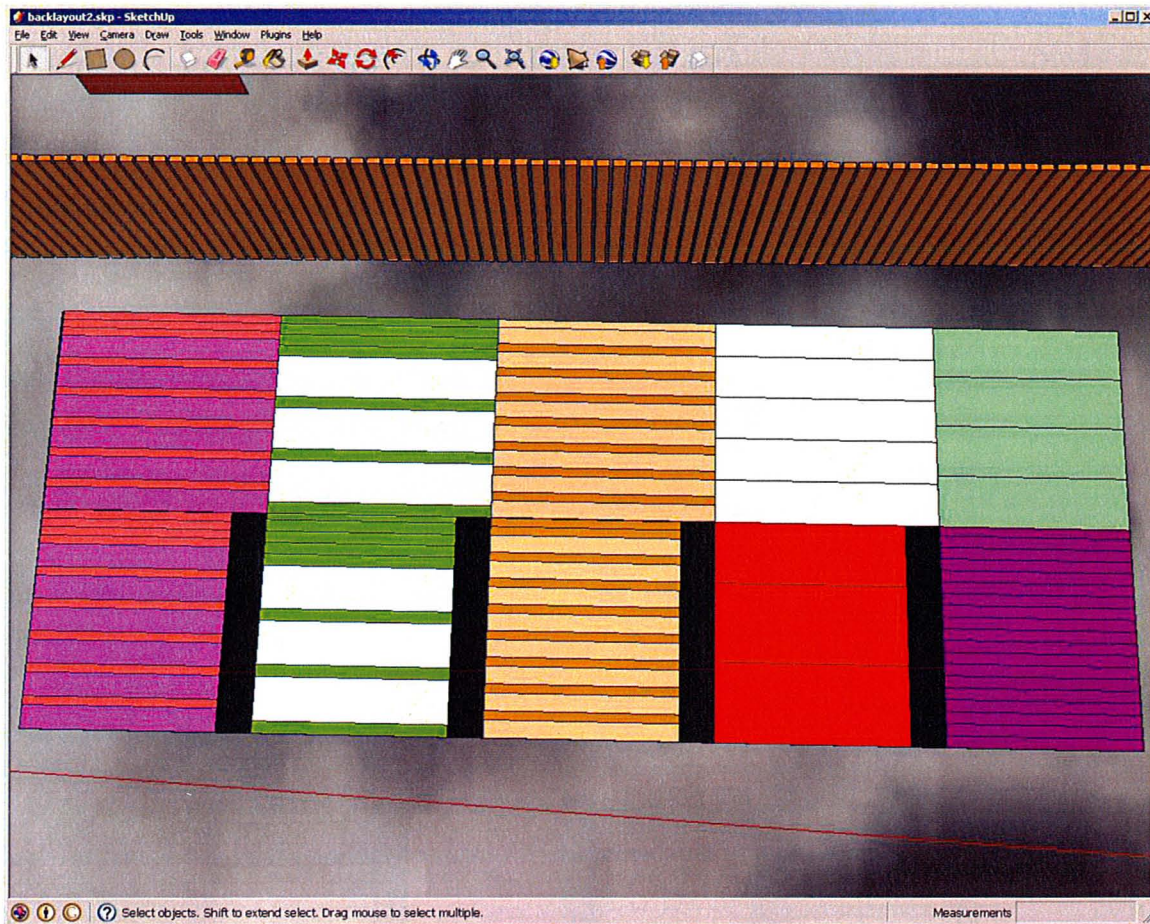


Figure 4.4 Backyard Garden: Trial 2. Largest maximum reaching length (36") and default/smallest path width (12"). Notice that each pair is represented twice as many times as a single vegetable.

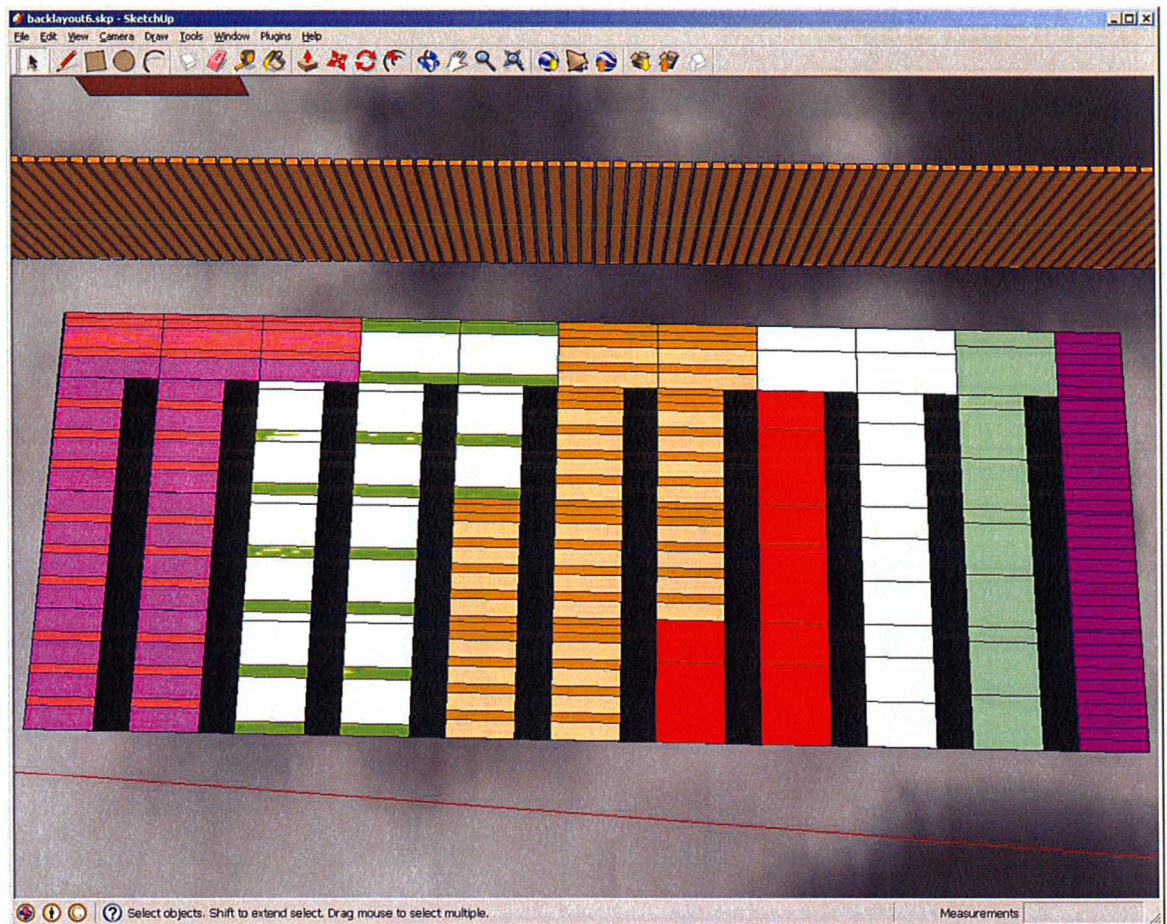


Figure 4.5 Backyard Garden: Trial 6. Smallest maximum reaching length (12") and smallest path width (12"). Notice that the areas beside the paths are split into 3 separate areas.

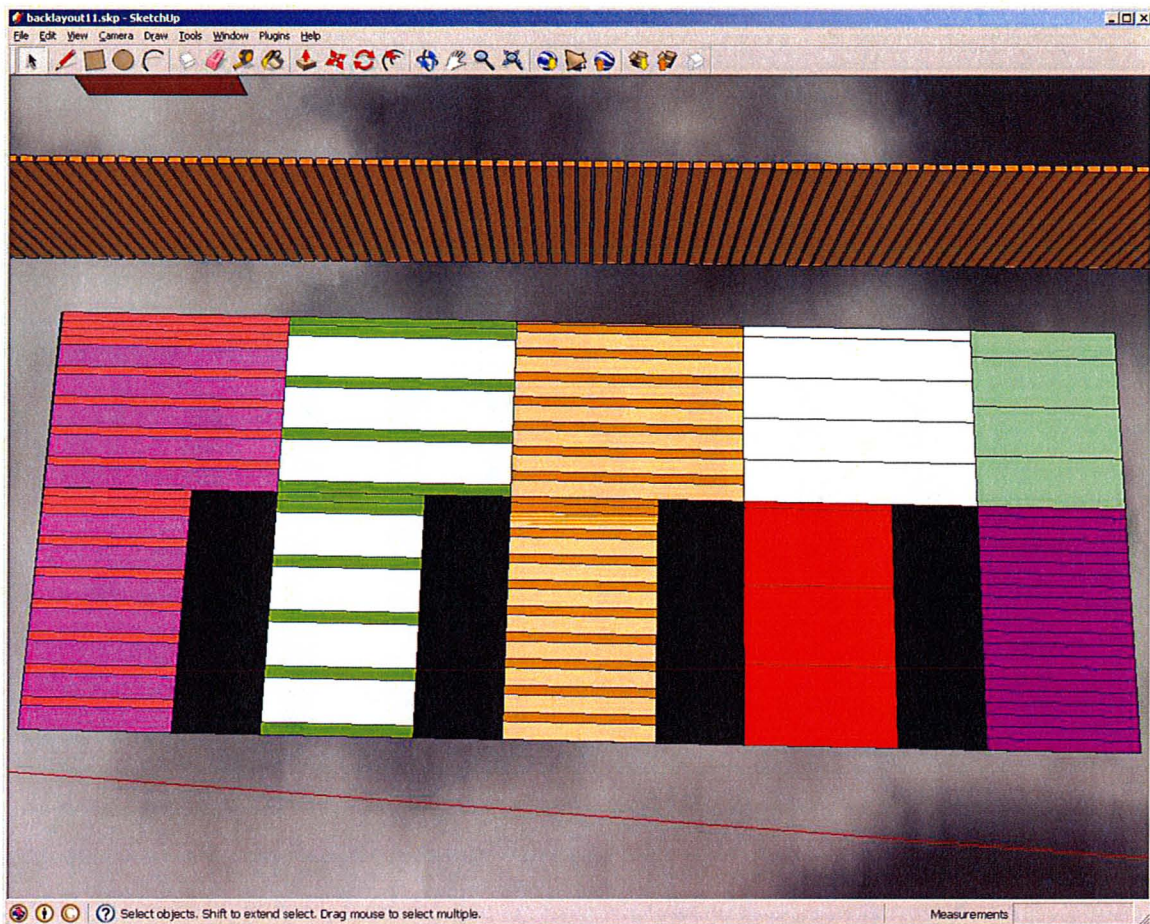


Figure 4.6 Backyard Garden: Trial 11. Default maximum reaching length (32") and largest path width (30"). Notice that pairs are represented twice as many times as a single vegetable; this is considered preferable.

4.2.3 Front Yard Garden

Table 4.2 includes documentation of the relevant inputs (maximum reach, path width, site conditions and vegetables) of each of the backyard trials as well as the trial number, (rounded) gardenable area (area of the rectangle minus path area), and any relevant notes. Trials one through eleven cover ranges of maximum reaching length and path width, changing one variable at a time. From trials 12 to 18, site conditions are varied; full sun and partial sun are used in the Sun Conditions category. Some representative screenshots of the trials are shown following the table.

Table 4.2 Front Yard Case Study

Trial	Max. Reach (in.)	Path Width (in.)	Site Conditions (Decorative, Sun, Type , Sowing)	Vegetables	Gardenable Area (ft ²)	Notes
1	32	12	All All All All	basil Brussels-sprouts cabbage(oriental) celery corn leek lettuce nasturtium pea potato	346	
2	28	12	All All All All	basil Brussels-sprouts cabbage(oriental) celery corn leek lettuce nasturtium pea potato	348	
3	24	12	All All All All	basil Brussels-sprouts cabbage(oriental) celery corn leek lettuce nasturtium pea potato	333	
4	18	12	All All All All	basil Brussels-sprouts cabbage(oriental) celery corn leek lettuce nasturtium pea potato	318	
5	14	12	All All All All	basil Brussels-sprouts cabbage(oriental) celery corn leek lettuce nasturtium pea potato	301	
6	12	12	All All All All	basil Brussels-sprouts cabbage(oriental) celery corn leek lettuce nasturtium pea potato	268	
7	32	14	All All All All	basil Brussels-sprouts cabbage(oriental) celery corn leek lettuce nasturtium pea potato	340	
8	32	16	All All All All	basil Brussels-sprouts cabbage(oriental) celery corn leek lettuce nasturtium pea	335	

9	32	18	All	potato	330	
			All	basil Brussels-sprouts		
			All	cabbage(oriental)		
			All	celery corn leek		
10	32	24	All	lettuce nasturtium pea	315	
			All	potato		
			All	basil Brussels-sprouts		
			All	cabbage(oriental)		
11	32	30	All	celery corn leek	299	
			All	lettuce nasturtium pea		
			All	potato		
			All	basil Brussels-sprouts		
12	32	12	Yes	basil leek lettuce	346	No antagonists present. Not all of original vegetable selection available in list.
			All	nasturtium		
			All			
			All			
13	32	12	Yes	basil leek lettuce	346	Not all of original vegetable selection available in list.
			All	nasturtium beet		
			All	bean(bush) cabbage		
			All	fennel peppers amaranth		
14	32	12	All	basil Brussels-sprouts	346	
			Full	cabbage(oriental)		
			All	celery corn leek		
			All	lettuce nasturtium pea potato		
15	32	12	All	lettuce parsley beet	346	No antagonists present. Not all of original vegetable selection available in list.
			Partial	borage chard		
			All	rosemary		
			All			
16	32	12	All	basil Brussels-sprouts	346	Celery not available in vegetable list.
			All	cabbage(oriental)		
			Annual	corn leek lettuce		
			All	nasturtium pea potato		
17	32	12	All	beet borage chard	346	No

			Partial All Outdoors	lettuce raspberry parsley rosemary		antagonists present. Not all of original vegetable selection available in list.
18	32	12	All All All Indoors	basil Brussels-sprouts cabbage(oriental) celery corn leek	346	Not all of original vegetable selection available in list.
19	32	30	All All All All	basil bean(bush) bean(pole) broccoli cabbage carrot chard eggplant fennel kale thyme tomato pea peppers	299	Chard and thyme not included in layout: more vegetable pairs than areas.

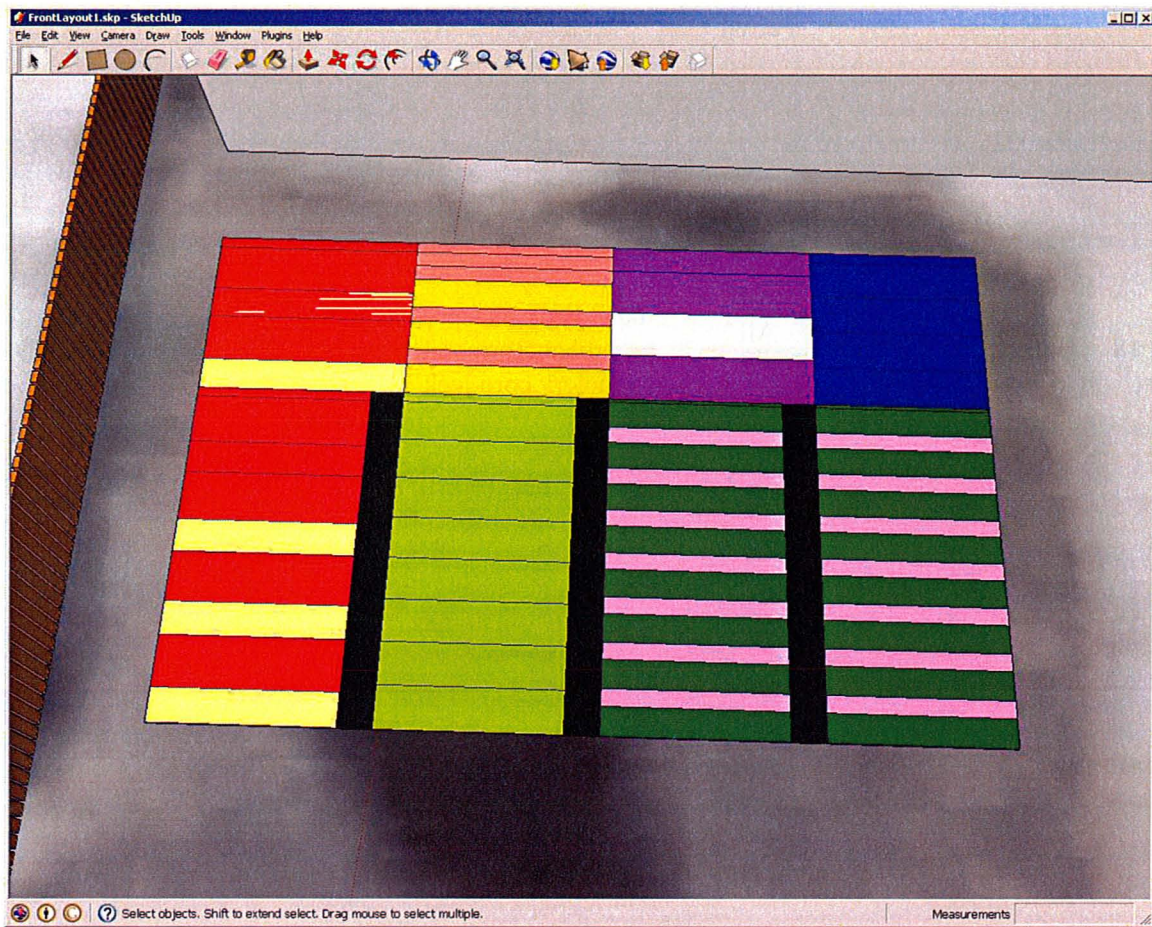


Figure 4.7 Front Yard Garden: Trial 1. Largest/default maximum reaching length and smallest/default path width (32" and 12", respectively).

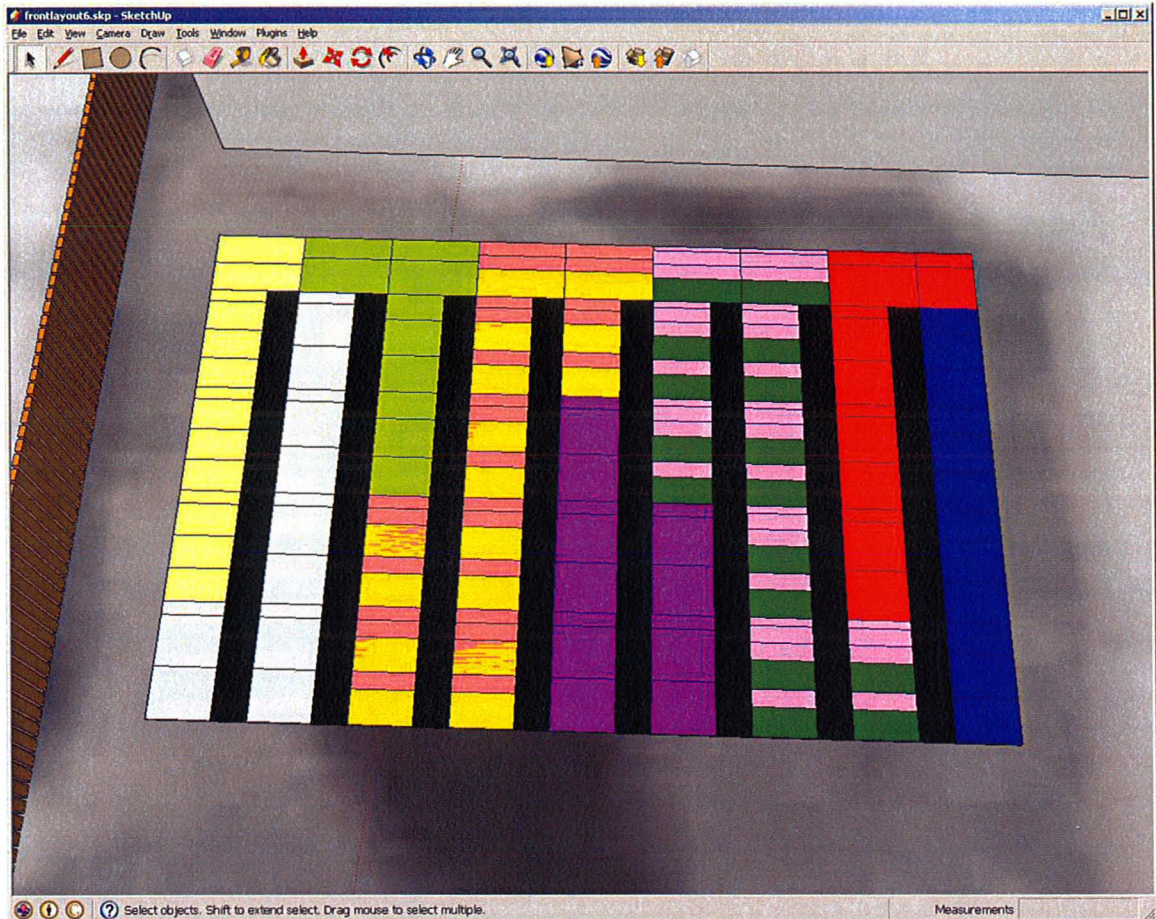


Figure 4.8 Front Yard Garden: Trial 6. Smallest maximum reaching length (12") and smallest/default path width (12"). Notice that the areas beside the paths are split into 4 separate areas.

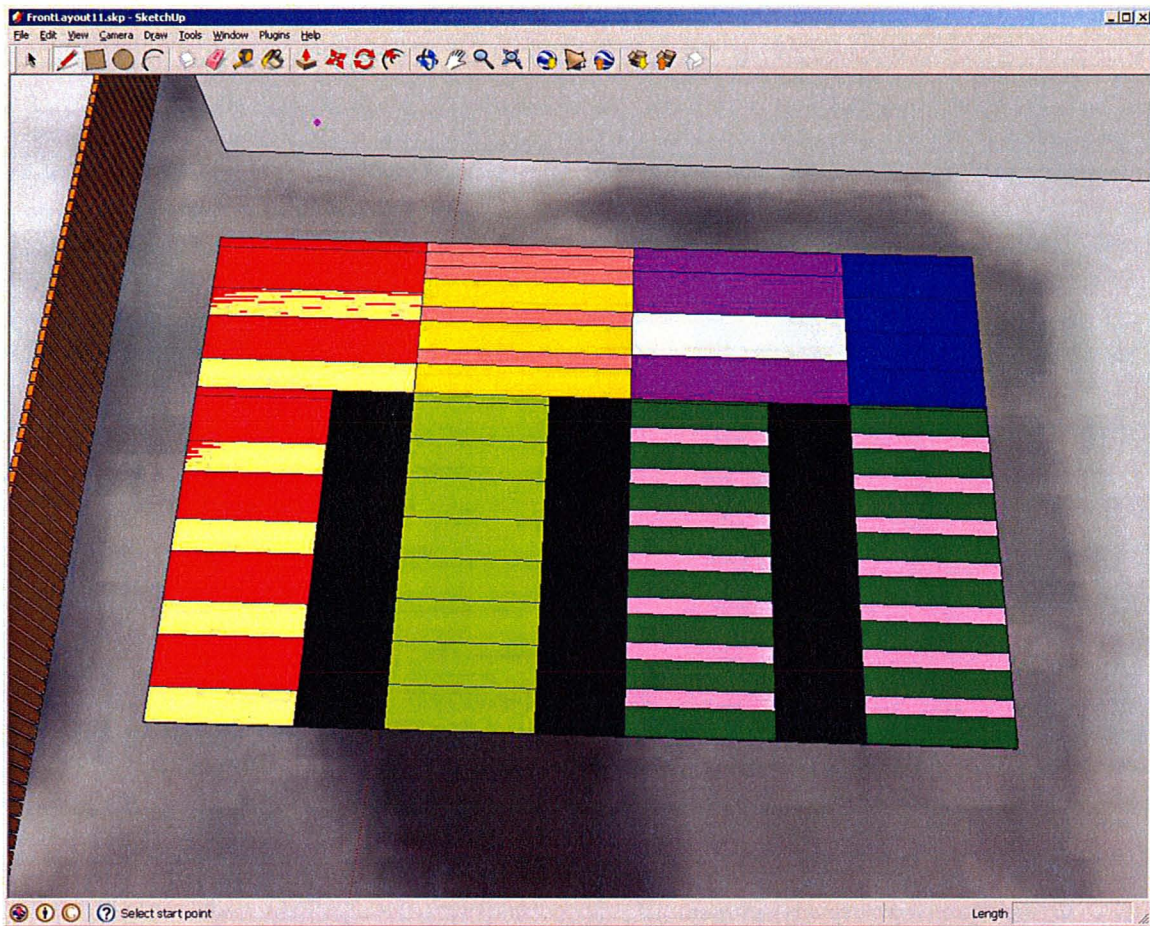


Figure 4.9 Front Yard Garden: Trial 11. Default maximum reaching length (32") and largest path width (30").

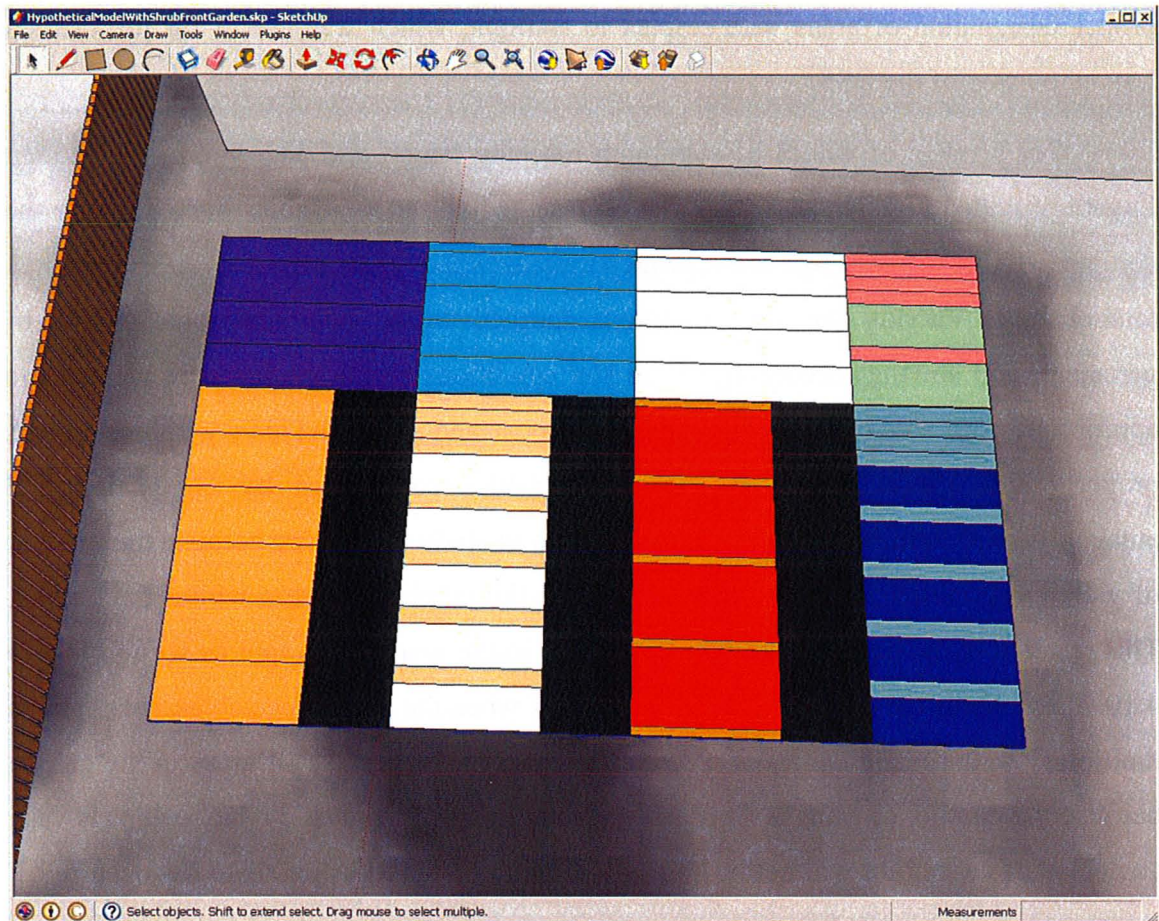


Figure 4.10 Front Yard Garden: Trial 19. Default maximum reaching length (32") and largest path width (30"). Fourteen vegetables were chosen to illustrate a limitation of the plugin; only twelve vegetables are present in the graphical layout. Two vegetables were not included because the number of paired vegetables in **pairlist** is more than the number of areas in the garden.

4.2.4 Comments Regarding the Hypothetical Application

Modelling the surrounding objects of the land parcel proved to be relatively straight-forward and required very little skill with SketchUp. The Measuring Tape tool was indispensable for approximating the lengths and widths of surrounding objects for the purposes of modelling. The layered image modified using GIMP was appropriate for showing different levels of sunlight cast. The use of the layered image allowed for the

proper designation of areas with respect to sunlight, which was useful for the Garden Layout portion of the DSS.

For a range of values of maximum reaching length and path width, the Garden Layout provided a graphical output that represented all ten vegetables picked within the confines of the garden and were properly constrained by companion and antagonist relationships. Varying the site conditions (attractiveness, sun conditions, annual or perennial, and sowing) according to the sun conditions of the garden area yielded the appropriate vegetable options from which to choose and also provided a graphical output properly showing at most ten chosen vegetables constrained by companion and antagonist relationships. One case was shown to exclude vegetables because the number of paired vegetables compared was larger than the number of areas within the garden. This resulted in the exclusion of two vegetables in the graphical output as seen in Figure 4.10. This reveals the plugin to have a limitation: when the list of vegetables, **pairlisto**, is multiplied so that the quantity of elements is equal to the number of areas in the garden because the number of areas is fixed and the variability of vegetable relationships is vast, exclusion of vegetables will not be known until multiplication of **pairlisto** has occurred.

4.3 Actual Case Studies

In these case studies three different users were asked to apply the DSS and give comments and suggestions about the instructions and/or the functionality of the DSS. The users chose their own residence in each case. Information and results of each case study are documented in this section in tabular and graphical formats with more detail given in the text, if needed.

4.3.1 DSS Results for User 1

This section includes a tabular overview of each user's inputs and any relevant notes followed by the graphical output of each user's trials.

Table 4.3 User 1's Input and Notes for DSS Trials

User	Maximum Reach	Path Width	Site Conditions (Decorative, Sun, Type, Sowing)	Vegetables	Gardenable Area	Notes
1	75cm	30cm	All Partial Annual Outdoors	Borage Calendula Lettuce	8m ²	Only three vegetable options available.
1	75cm	30cm	All Partial All All	Asparagus Blackberry Blueberry Beet Lettuce Raspberry Rosemary Parsley	8m ²	Asparagus, raspberry, and parsley areas not present in graphical layout (Figure 4.13).

The first user of this DSS applied both the Shadow Analysis plugin and the Garden Layout plugin using a Google Earth image of their current residence in Hamilton. The entire session took 2h 40min. The user had no prior experience with Google SketchUp or GIMP. The result of the user's Shadow Analysis is shown in Figure 4.11. The user chose his backyard as the location for a garden plot. The graphical outputs from two Garden Layout trials are shown in Figure 4.12 and Figure 4.13. An error occurred in the second trial when blank areas (shown as black), representing vegetable planting areas, occurred in the graphical output of Garden Layout. This was due to the inappropriate designation of areas to pairs of vegetables whose spacing was larger than the length of the area. The code has been altered in an effort to remedy this problem such that if the vegetable's spacing is checked before it is paired and if its spacing is larger than the area lengths, it will not be paired. Similarly, each spacing of a potential pair of vegetables is added together and if this combined length is larger than the length of either of the gardening areas, they will not be paired. Also, it can be seen in Figures 5.8 and 5.9 that the user failed to draw and input a rectangle (all corners 90 degrees) for use in the Garden Layout plugin. This does not seem to have altered the outcome of the plugin from that of a rectangle.

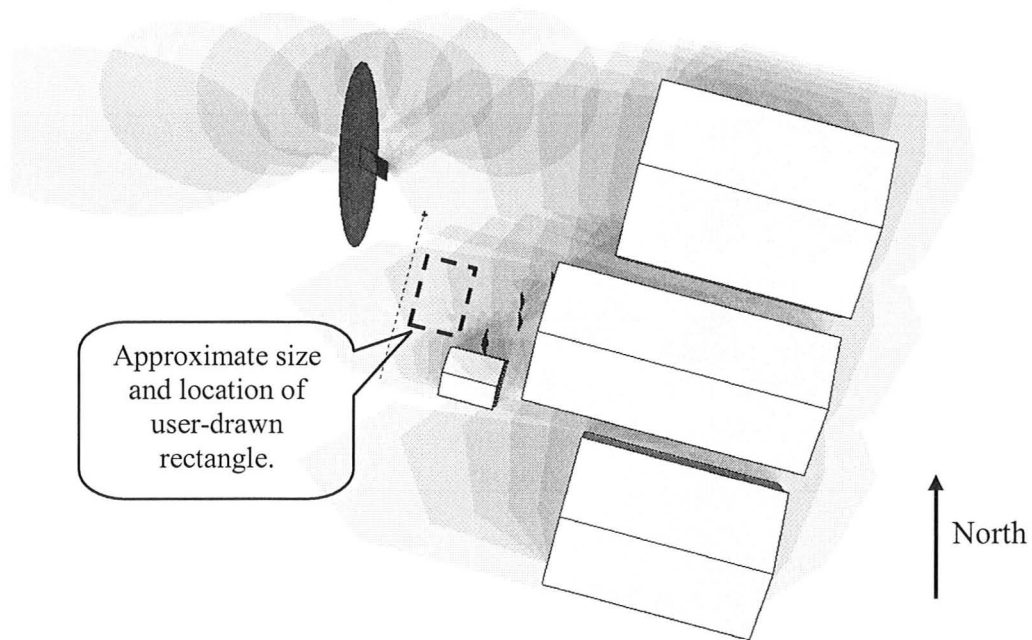


Figure 4.11 User 1: Shadow Analysis. The shadows represent the 24th of August, 2010.

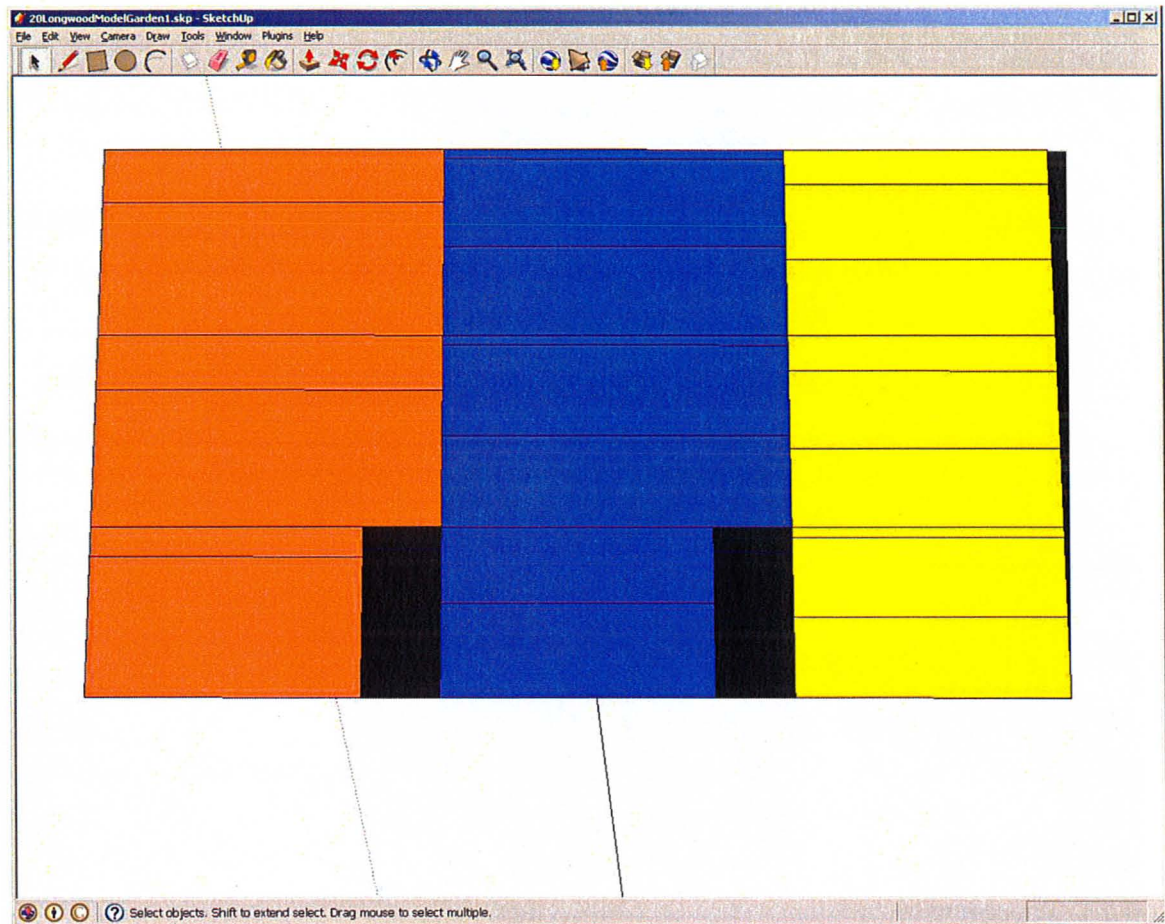


Figure 4.12 User1: First Trial of Garden Layout

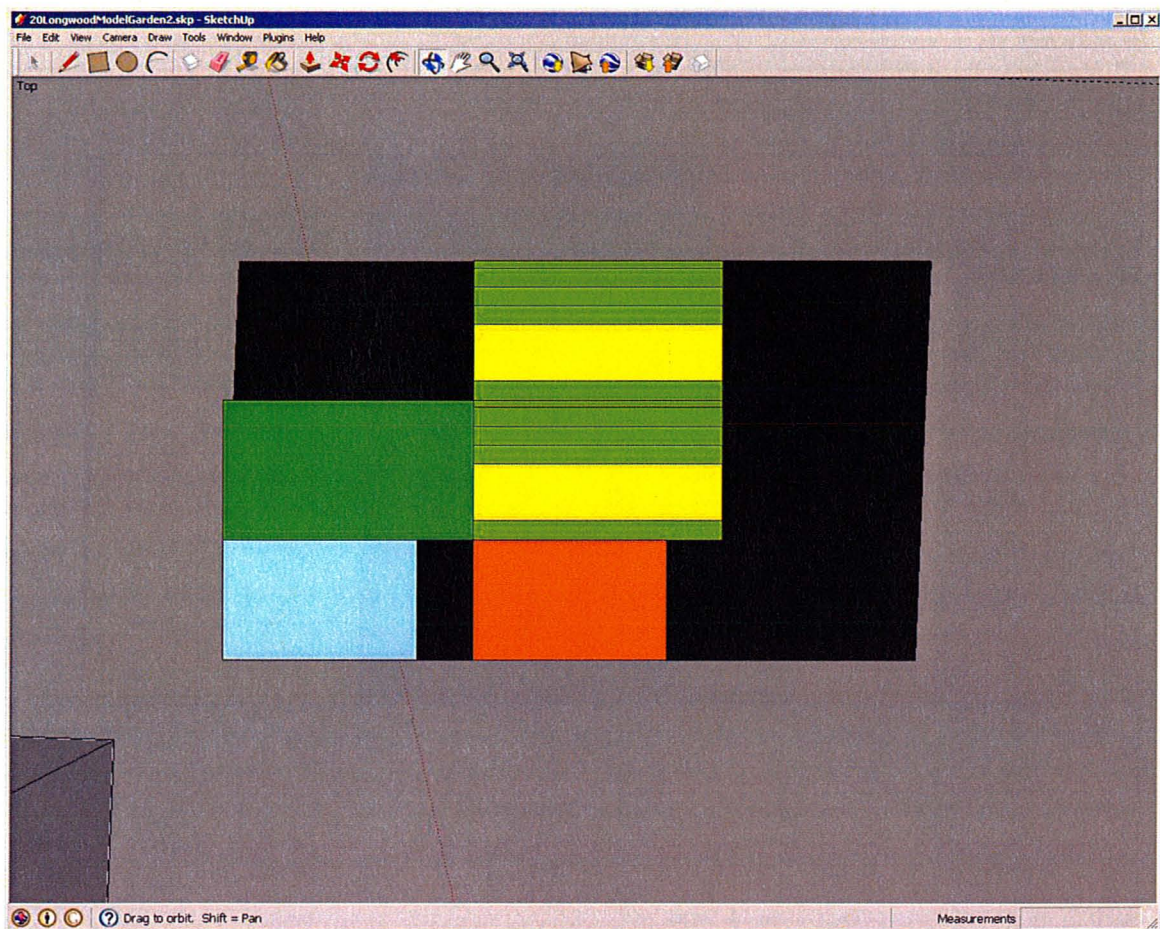


Figure 4.13 User 1: Second Trial of Garden Layout

4.3.2 DSS Results for User 2

Table 4.4 User 2's Inputs and Notes for DSS Trials.

User	Maximum Reach	Path Width	Site Conditions (Decorative, Sun, Type, Sowing)	Vegetables	Gardenable Area	Notes
2	50cm	30cm	Yes Full Annual Outdoors	Sunflower Bean(various) Lettuce	47m ²	Side areas are not split into more areas, but in this case it is inconsequential.
2	50cm	30cm	All Full Perennial/ Biennial Outdoors	Blackberry Raspberry Blueberry Rosemary Garlic	47m ²	The ratio of the areas beside and above the rows are larger than four, the program had previously only dealt with ratios of less than four.

The second user of the DSS also used both the Shadow Analysis plugin and the Garden Layout plugin, using a Google Earth image of their current residence in Hamilton. The entire session took 1h 35min. This user had no prior experience with Google SketchUp or GIMP. The result of this user's Shadow Analysis is shown in Figure 4.14. He chose his entire front yard as the location for a garden plot. The graphical outputs from two Garden Layout trials are shown in Figure 4.15 and Figure 4.16. An error occurred in both trials, though only apparent in the second, where the areas beside the paths were not split into multiple areas even though they are more than four times the area of the areas above the paths. This was a limitation of the script where it would only split areas that were between three and four times larger into three areas and beyond this, the areas were not split. The script has now been changed so that it accommodates areas that are up to five times the area of the areas above the paths by splitting the area into four. Beyond this the area is simply split into four.

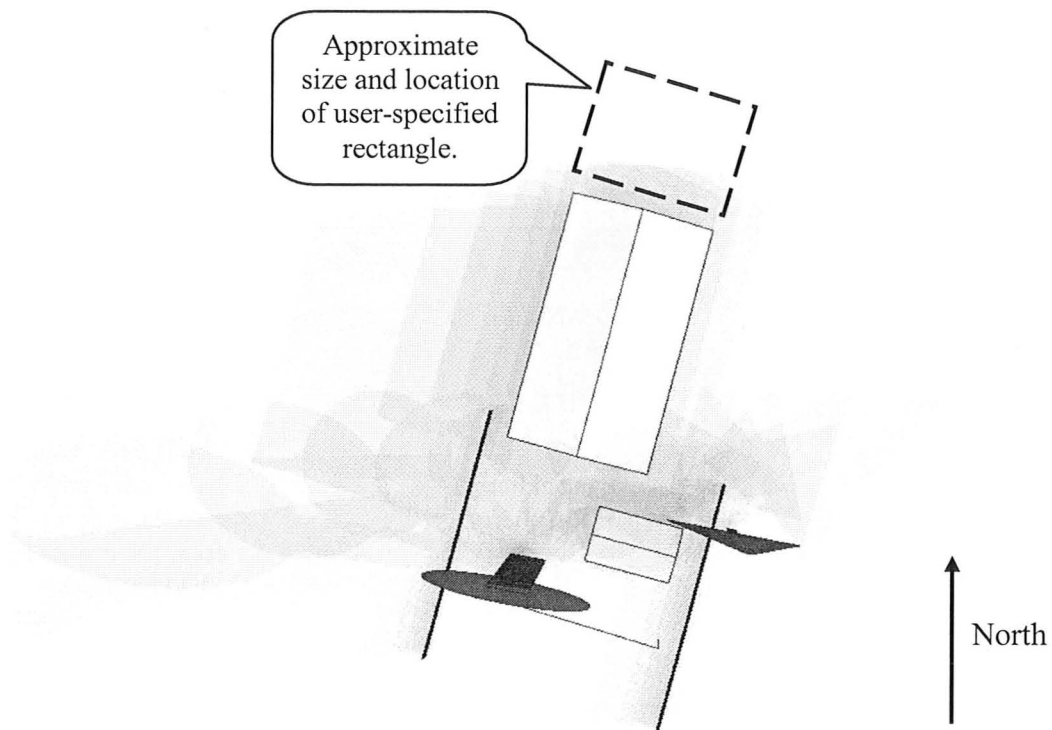


Figure 4.14 User 2: Shadow Analysis representing August 24th, 2010.

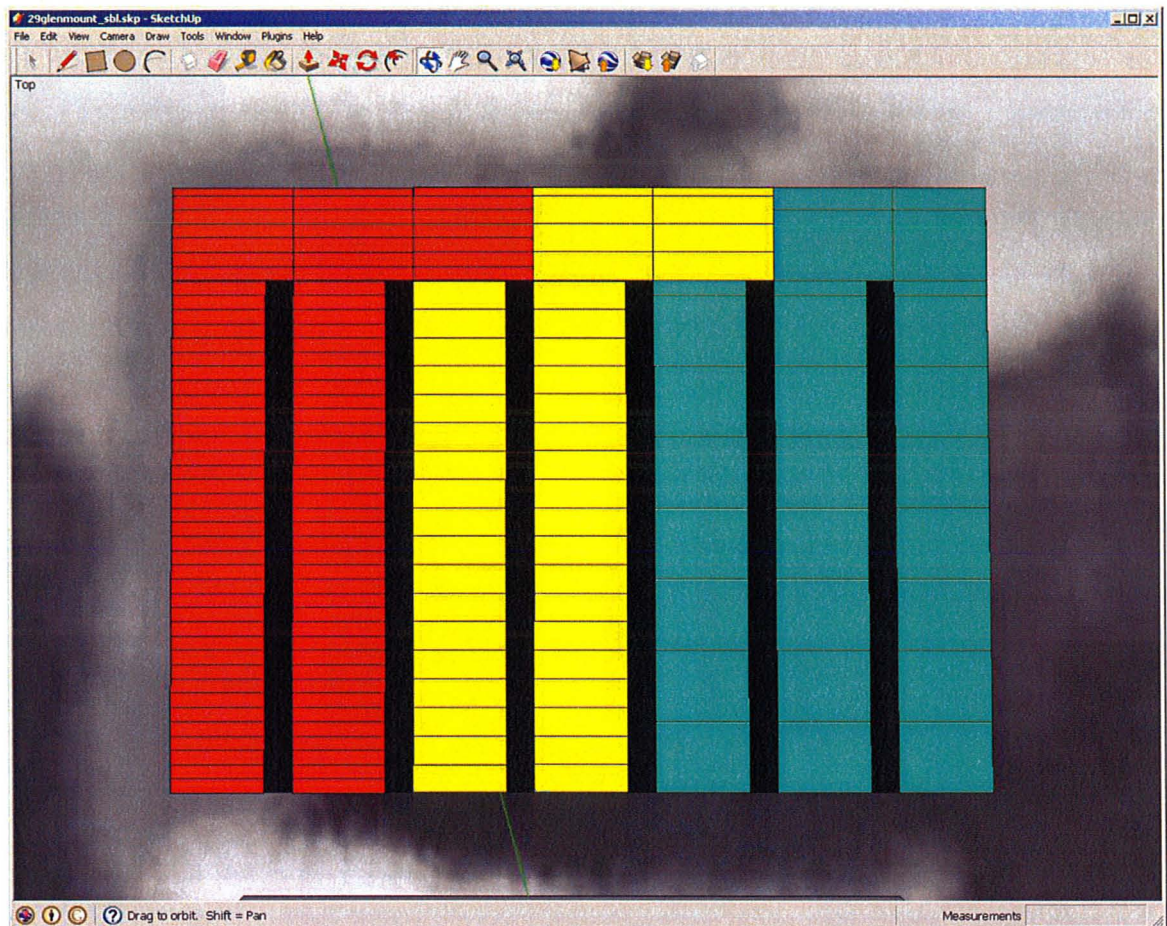


Figure 4.15 User 2: First Garden Layout Trial.

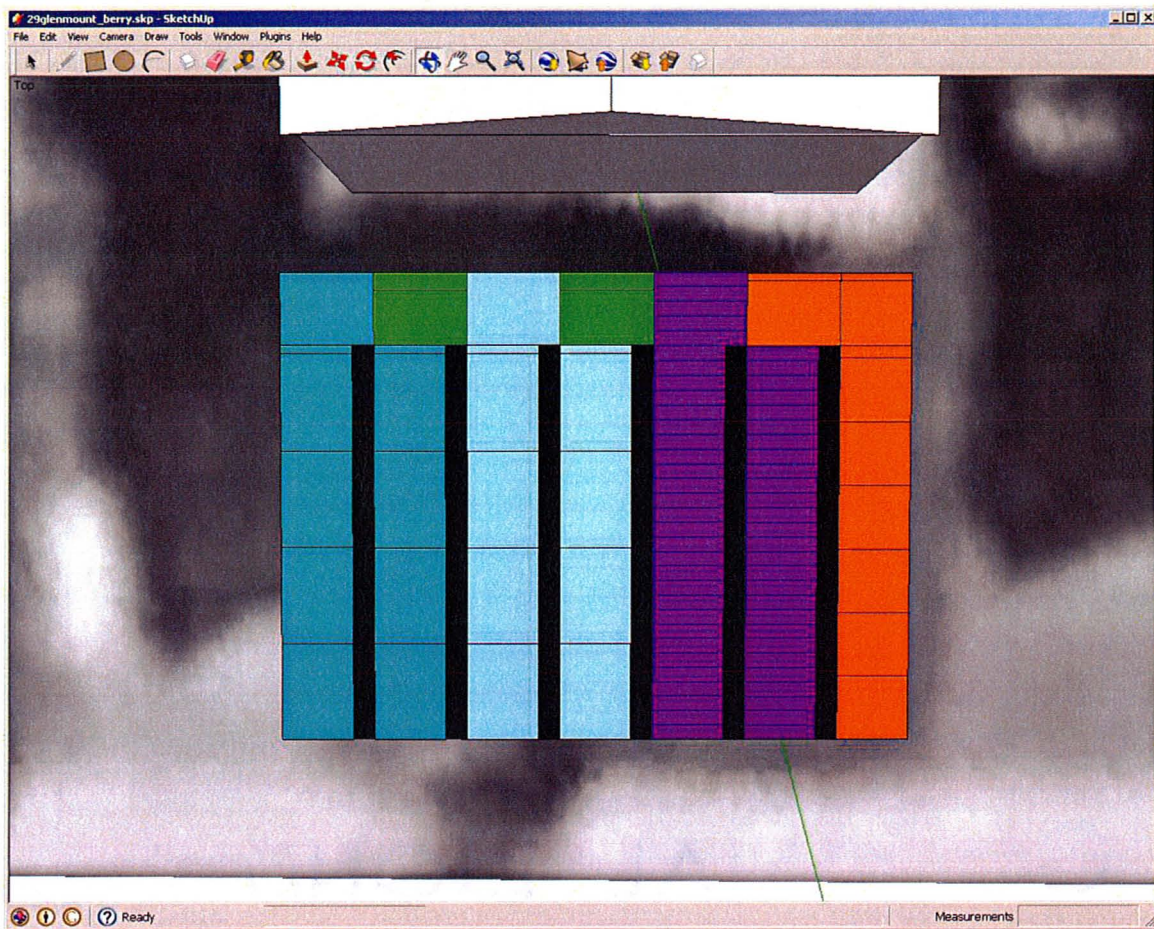


Figure 4.16 User 2: Second Garden Layout Trial. Notice the areas beside the path are not divided into smaller areas.

4.3.3 DSS Results for User 3

Table 4.5 Details of User 3's Garden Layout Trials.

User	Maximum Reach	Path Width	Site Conditions (Decorative, Sun, Type, Sowing)	Vegetables	Gardenable Area	Notes
3	32in	12in	All Full All Outdoors	Basil Beet Parsley Quinoa	64ft ²	Most of the area of the plot taken up by the row. Foot of garden is less than 1m long.
3	32in	12in	All Full All Outdoors	Beet Quinoa Asparagus	70ft ²	Twice reaching length is more than the length. This is a limitation of the plugin.
3	48cm	25cm	All Full All Outdoors	Beet Quinoa Asparagus	6m ²	Changed reaching length to be half of the length of the garden.

The third user of the DSS used the Garden Layout plugin without the use of the Shadow Analysis plugin using a Google Earth image of their current residence in Hamilton. The entire session took 50min. and three trials of the Garden Layout plugin were accomplished. This user had no prior experience with Google SketchUp. The graphical outputs of the three trials are shown in Figure 4.17, Figure 4.18, and Figure 4.19. She chose a small area (6.066m x 0.968m) at the side of her house as the location for a garden plot. The first trial used the smallest dimension (< 1m) as the width of the garden. Because of the 12in. (30.48cm) path width chosen, this left less than 35cm on each side for planting. This configuration encouraged the user to try the plugin again using the long dimension (> 6m) as the foot of the garden.

An error occurred in this second trial: the graphical output was indecipherable because of overlapping areas, which also had the effect of showing a larger length of the

plot. Because the input reaching length was 32in. (81.28cm), the length of the garden in the graphical output was 1.63m (twice the reaching length,) which is obviously larger than the length of the plot (0.968m). This error occurred because the script assumes the user will not model a garden that is less than twice their reaching length long. This continues to be a limitation of the DSS, but a warning to the user has been added.

Recognizing the error and with the desire for the user to complete her previous attempt, the researcher suggested to input a reaching length that is at most half of the length of the rectangle. Using the same vegetables and configuration but an altered reaching length (and path width due to metric units), the graphical output gave an appropriate layout, which can be seen in Figure 4.19.

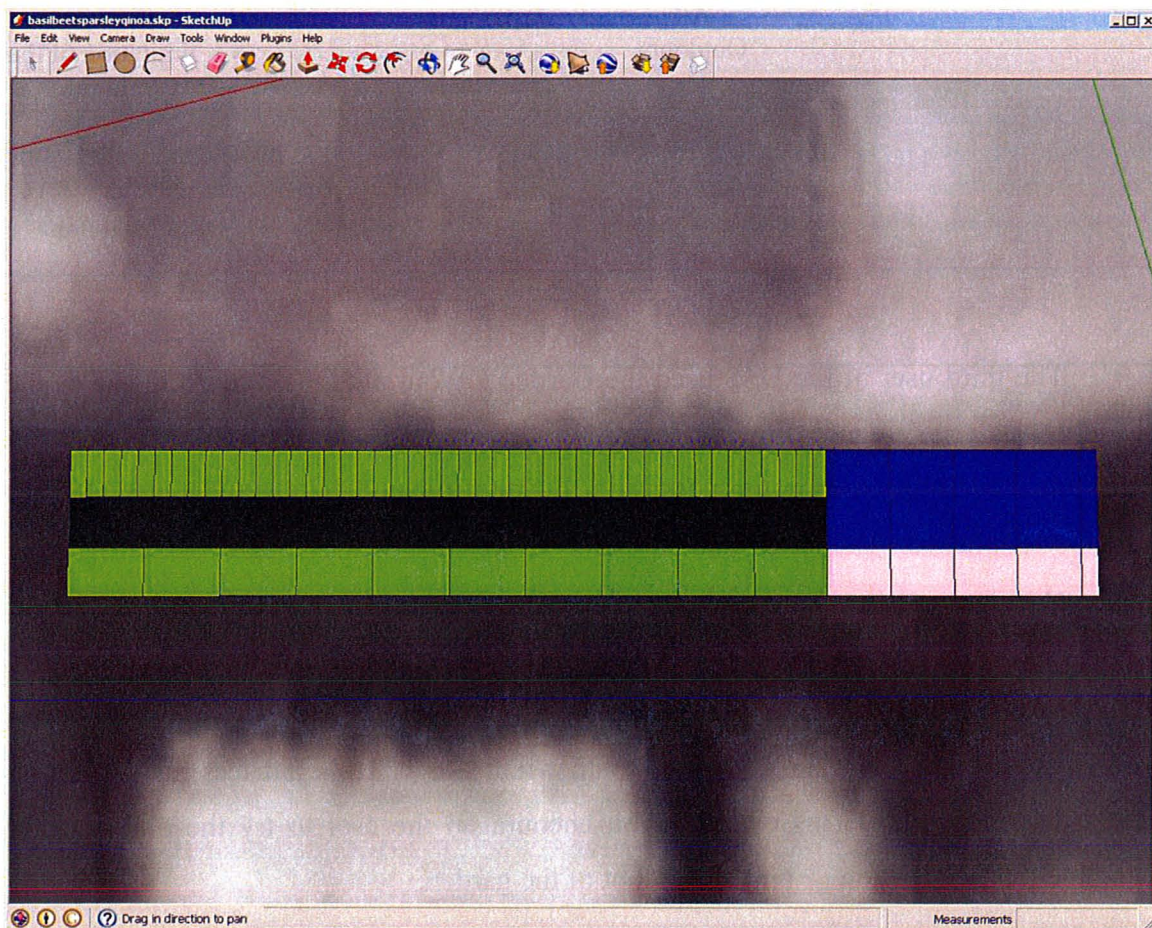


Figure 4.17 User 3: First Garden Layout Trial.

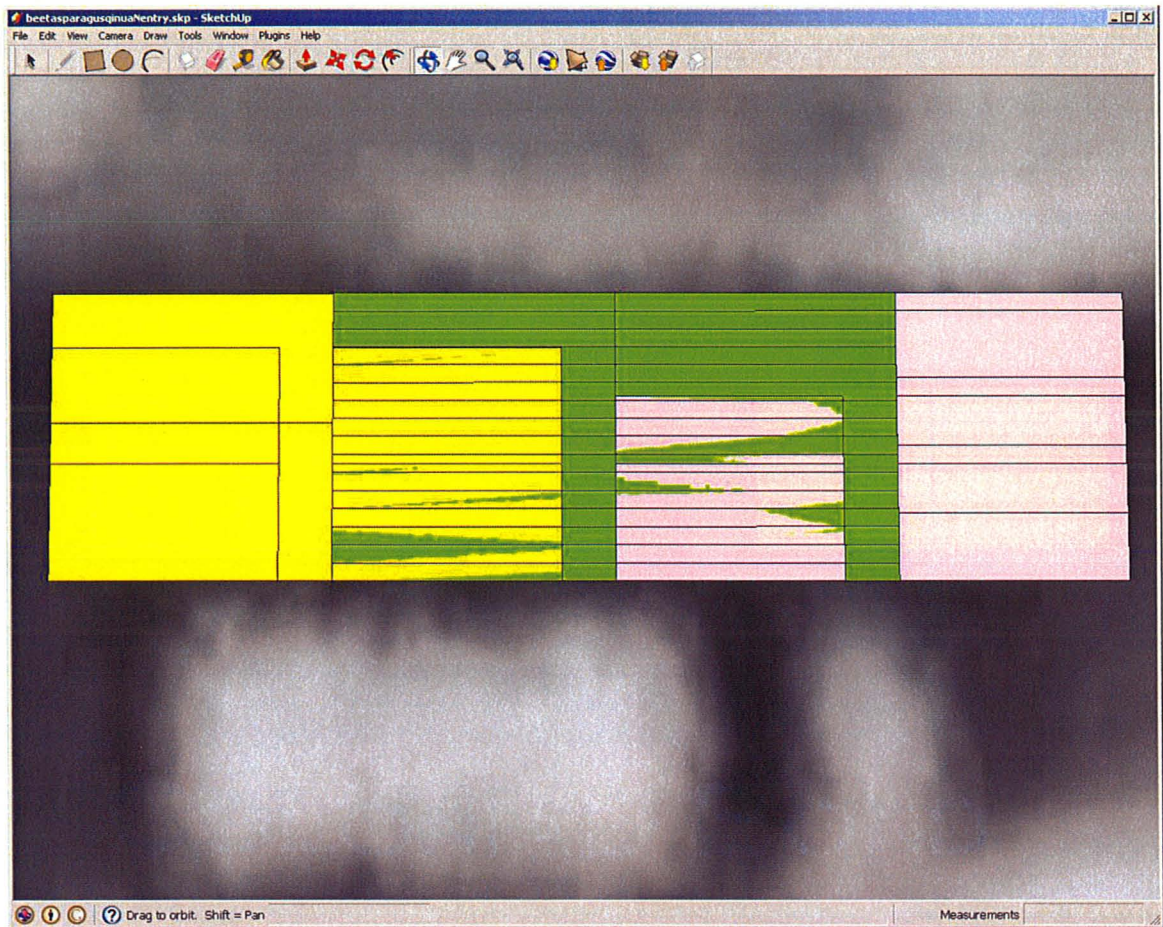


Figure 4.18 User 3: Second Garden Layout Trial. Notice the relative dimensions; the length (shorter of the dimensions) is larger than the other trials.

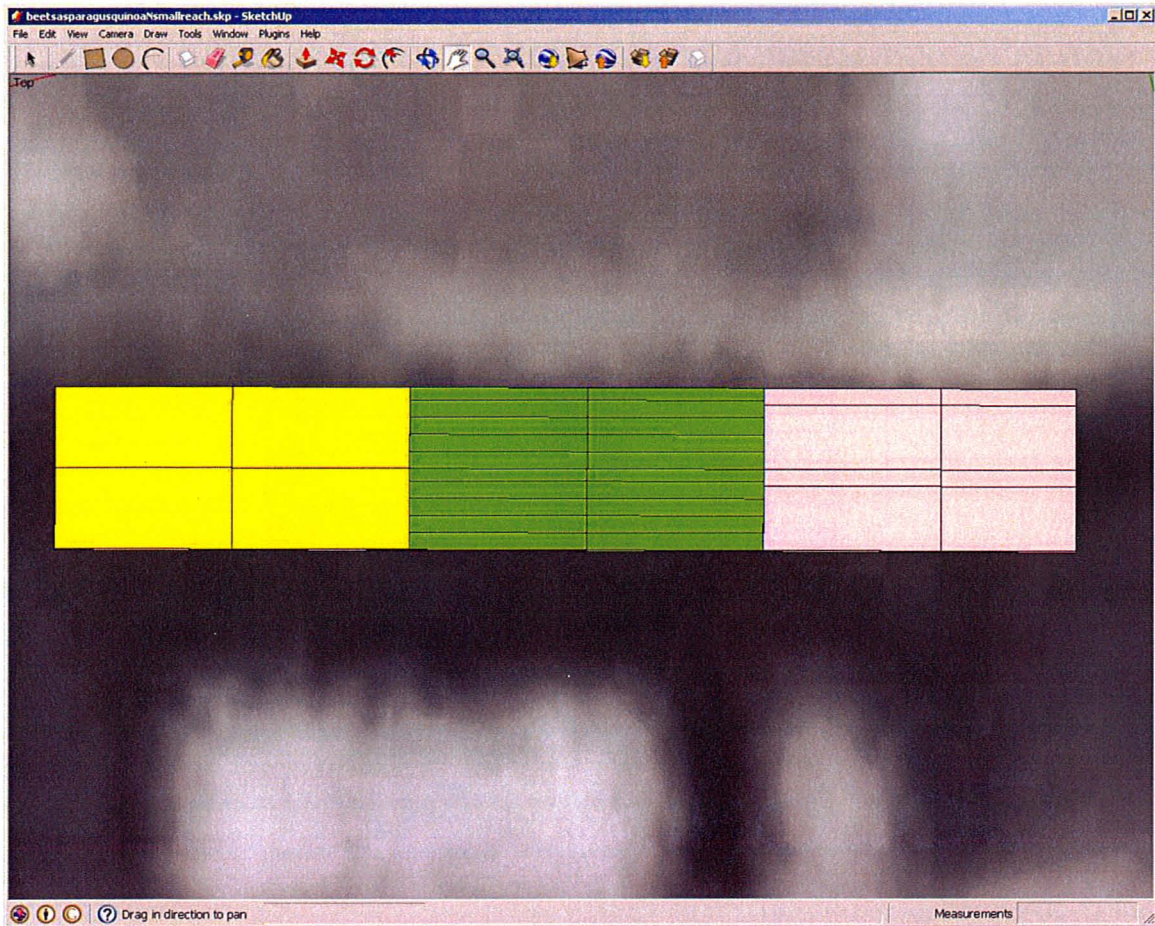


Figure 4.19 User 3: Third Garden Layout Trial.

4.3.4 Comments and Progress

Table 4.6 includes all comments made in the actual case studies. The comments are divided into those pertaining to instructions and function of the plugins. The last column in the table is dedicated to the progress of any changes that can be made based on the related comment.

Table 4.6 Compilation of Actual Case Studies

User	Instructional Comments	Functional Comments	Progress
1	More detail on how to rotate and change angle of view in Google Earth		The user instructions now include a brief description of some key tools of Google Earth and how to use them.
1		Components should be selected upon creation.	Components in the Shade Tools submenu are the only object selected upon their creation.
1	Let user know that Move tool rotates components and Rotate tool rotates everything else.		Distinction between rotations using the Rotate tool vs. Move tool captured in Useful Tools section of instructions.
1	Select Google Earth image (highlights red) before activating Shadow Analysis.		Instructions to select Google Earth image before using Shadow Analysis now included.
1	When saving layers in GIMP, merge visible layers.		Instructions on the details of saving an image in GIMP are now included.
1	Explain 'bottom edge' and 'foot' of garden.		Explanation of bottom edge and foot of garden are now included.
1	Explain how and when to use the camera tool.		Camera included in Useful Tools section and its use recommended in appropriate portions of instructions.
1		Useful error message when more than 3 items are selected.	Error message is the same.
1	Make clear that if fence is chain-link, the fence component may not be appropriate due to shadow		Chain-link fence considerations and additional suggestions about property lines have been included.

	considerations.		
1	Use screenshots from an example to help ease of use in instructions.		A diagram of parts of a garden is now included in instructions.
1		Difficulty keeping objects on the x-y plane.	More descriptive account of how to move objects up and down now included.
1,2		Deciduous and Coniferous need to be more descriptive.	Descriptions now included in brackets beside Deciduous and Coniferous, respectively.
1		In second run, some vegetable faces did not appear in final layout.	Inappropriate designations of vegetables remedied in script.
2	Make it clear to check that objects are on the ground plane after model is finished.		Include instructions to check that modeled objects are on ground plane.
2	Include that view does not matter before using Shadow Analysis.		Comment about view in SketchUp before using Shadow Analysis included.
2		Wanted an easier way to get information out of Shadow Analysis.	Note about the option to make an image of one month at a time now included.
2	Save your shadow analysis as some other name, if not, open old version and click 'yes' to reverting back and 'no' to opening the auto-saved version.		Suggestion about how to save shadow analysis now included in instructions.
2	Garden Layout step 3: save work as using a different file name.		Suggestion about how to save work included.
2	Selecting three items in step 5 not clear.		More descriptive instruction on how to select and what to select included.
2		Areas beside the paths did not split into multiple areas. The bottom to top area	Script has been modified to accommodate a ratio of up to 5. Above this

		ratio is >4.	the area is split into four.
2		Have the ability to remove vegetables that have been picked.	Script has not been modified. User will have to restart Garden Layout.
2,3		Have a checkbox for all the vegetables (which would make the remove function moot).	Checkbox format is not available in SketchUp's API.
3	When user does not use Shadow Analysis: make clear that doing steps 1 and 2 of Shadow Analysis instructions is required to start the Garden Layout.		Following steps 1 and 2 of Shadow Analysis included at the beginning of Garden Layout.
3		Is the shattered-looking face of rectangle normal?	Shattered appearance of rectangle face explanation included.
3	Include cautionary statement explaining that the rectangle face needs to be selected in order to rotate rectangle with Rotate tool.		Notification of selecting face to be able to perform rotations with the Rotate tool now included in instructions.
3	Explain: Decorative Edibles.		Description of decorative edibles included.
3	Explain 'All' in the context of the site condition dropdown menus.		'All' is explained in each site condition option.
3	Explain/change antagonist message.		Antagonist message changed to inform the user that all groupings of plants can be moved to their liking.
3		The inputted reaching length is larger (or twice the inputted reaching length) than the length of the garden plot. This causes the graphical	This is noted in the instructions as a warning to users.

layout to be
indecipherable.

The first two users took 1h 50min. and 1h 35min. respectively. They each modeled approximately the same scenario and applied the Garden Layout twice. It is reasonable to assume that the augmentation of the instructions between users contributed to some of the difference in time taken to complete both the Shadow Analysis and Garden Layout. Each of the first two users were generally satisfied with the outcome of the Shadow Analysis and provided suggestions pertaining to functional aspects to increase the ease of use of the Shadow Analysis plugin. User 3 took a total of 50 minutes to complete three trials of the Garden Layout. All users were generally satisfied with their first outcome of Garden Layout and each user subsequently requested to apply the Garden Layout a second time. Users 1 and 2 decided to conclude their DSS session after two trials of Garden Layout; time restrictions were a contributing factor. The third user was dissatisfied with her second trial (due to an indecipherable graphical layout) and continued to attempt a third trial which produced satisfactory results.

Using the suggestions and recommendations from users of this DSS prototype allowed for the strengthening of the DSS as well as its related instructional material. This prototype DSS has the potential to be modified even further through feedback from future users. The following chapter will outline recommendations for the prototype DSS, its use in future work, and conclusions drawn from this research.

Chapter 5 Conclusions and Recommendations for Future Work

5.1 Summary

This decision support system serves to assist urban and suburban residents, in Southern Ontario, interested in producing a portion of food for their own consumption by helping them to plan a suitable garden arrangement on their land parcel. This thesis describes the development of a prototype DSS consisting of two modules: the first module for characterising the sun conditions on the land parcel (referred to as Shadow Analysis), and a second module which generates a garden layout from user-selected inputs (Garden Layout). The Shadow Analysis module assists users in learning about the sunlight conditions of their land parcel. The Garden Layout module may be used with the Shadow Analysis output or can be executed on its own. This second module takes in the user-specified inputs of site characteristics to help the user select from a variety of vegetables and also uses inputs from the user such as reaching length and path width to provide a layout that satisfies the user's working condition preferences. The result of this exercise is a graphical representation of a garden layout that can be used for developing garden plans.

The objective of this thesis has been to communicate the motivations behind the need of individuals to increase their resilience by providing food for themselves from their own backyards and to provide a prototype DSS to address this need. This chapter is intended to provide conclusions from this research, outline potential amendments likely to improve the prototype DSS, and suggest future work that is related to the prototype DSS development.

5.2 Conclusions

Growing food in urban areas can act as a viable source of food for the urban or suburban resident. Utilising yard space for food production in urban and suburban areas is becoming more and more common. Positive impacts to individuals and communities result from local food production for individual consumption.

The closer food is grown and processed to the person consuming it, the less GHGs, fossil fuels, and resources it takes to move it to that consumer. Growing produce that would typically be purchased from a grocery store offsets the cost of that produce. Gardening provides light exercise for gardeners. Growing produce ensures consumption of nutrients from vegetables to the grower's family. Also, individuals involved in community gardening increase social connections because of time spent with others at the community garden.

With respect to the way individuals typically acquire vegetables, conventional farming in North America is not a sustainable method of food production. Reduced input and organic farming is more sustainable than conventional, industrial-style farming and is more likely to be used on smaller scales such as backyard gardening. Southern Ontario is made up of a high percentage of arable land that is being used inefficiently for residential and industrial uses. By increasing the amount of food grown in southern Ontario, residents can decrease their dependence on imported food. Provincial and/or federal policies around more sustainable agriculture can improve the health of the environment. Regarding small-scale food production, using an urban or suburban yard for food production is more sustainable and productive than the typical ornamental lawn on an urban or suburban lot. Also, producing food on an urban land parcel, which may have otherwise come from the supermarket, increases biodiversity and reduces the need for food transportation.

Companion and antagonistic relationships are based on resource usage, pest control, and garden conditions; the relationships can be used to alter the yields of either or both of the plants in the relationship. The developed Garden Layout module uses these

relationships in its planning algorithm, which is an advantage over other garden planning software that has been reviewed.

Using free software in this DSS is a crucial step in making it more accessible to the public, though downloading may be unattractive as compared to web-based systems. Google Earth images with the correct orientation, latitude and longitude, etc, representing the to-scale land parcel are available for free use and importing into SketchUp. Google Earth is a suitable software for use in this DSS because it provides satellite images of urban areas with sufficient resolution to distinguish houses and smaller features, which are necessary to discern in the developed DSS. The Google Earth satellite image is a suitable template on which the user can model their land parcel. SketchUp is a suitable software for use in this DSS because it is capable of: 3-d modelling, casting accurate shadows, creating various 2-d images such as JPEGs, receiving inputs, importing Google Earth satellite images, generating graphical output, etc. GIMP is a suitable image manipulation software for this DSS because it provides the functions of image multiplication and variation of image opacity. GIMP lacks the ability to select multiple images/layers which is a limitation; this feature could allow for more image manipulation by the user, resulting in more information passed to the user. The web-based garden planning tools reviewed in this thesis offer Flash-based 'drag-and-drop' features that add to the ease of use for the user. The number of plants and their approximate yield in the user's garden plan are helpful pieces of knowledge for the user.

The Shadow Analysis module of this prototype DSS is an important learning tool for the user and this opportunity for characterizing the potentially various areas of sunlight incident on a specific land parcel is not available in existing garden planning software. Using the site conditions of the parcel to select a list of site-compatible vegetables is an important feature for the user and could be improved by adding more features such as the ability to select soil conditions. The Garden Layout module uses a suitable number of input boxes that will not overwhelm the user. Pre-defined components of simplistic shapes of typical residential parcels are considered to be a helpful feature for the user. Though the tree components are useful for creating simplistic representations of

trees, a component that is more representative of the infiltration of light through a tree's canopy may be more useful to the user. Gardening information was predominantly obtained from non-academic sources.

The hypothetical trials covered in this DSS showed that the developed system could accommodate a large range of input variation and yield appropriate graphical layouts. The ranges tested in the backyard and front yard trials were what the researcher found as reasonable ranges for path widths (12"-36") and maximum reaching lengths (12"-30"); these resulted in satisfactory graphical layouts. The backyard trials showed that many site conditions appropriate for the area (i.e. the backyard garden was in an area that had virtually no shade throughout the day, only full sun was used, but all other conditions were varied) resulted in a variation in the list of vegetables available for the specific site conditions; the range of site conditions used in the backyard trials resulted in satisfactory garden layouts. The front yard trials showed a range of site conditions appropriate for the plot (i.e. both full sun and partial sun were used as sunlight conditions) and resulted in almost all appropriate graphical outputs. The 19th front yard trial yielded a garden layout that does not include all vegetables, herbs, and/or flowers that were chosen because the number of areas within the garden is less than the number of the list of vegetables that has been paired.

The actual case studies showed the range of capabilities for the prototype DSS, with variations in types and dimensions of gardens and types of users. Users unfamiliar with the software used in this DSS provided constructive feedback leading to improvements in the function of the DSS and instructional material. An example of a functional recommendation made by a user and implemented is that User 1 complained of difficulty deselecting the previously selected component after the current one had been created; the Shadow Analysis components made in SketchUp are now the only entities selected upon their creation, which allows the user to immediately move the component. Some errors occurred in the graphical output during the actual case studies, and these have been remedied. Coloured areas absent from the graphical layout were remedied by checking the length of each vegetable's spacing requirements and combined potential

pairs spacing requirements against the length of each area within the garden. Areas beside rows did not split into areas similar to that of the areas above the rows and the code was thus modified to allow up to five areas beside the rows. The last graphical error that occurred was when the length of user 3's garden was shorter than twice the maximum reaching length chosen by the user; a warning in the instructions has been added to aid the user. These improvements may have led, in part, to the reduction in time taken to execute the DSS over the range of users.

The time taken for the users in the actual case studies to complete the Shadow Analysis and Garden Layout modules or solely the Garden Layout module was between fifty minutes and one hour and 50 minutes. The researcher considers this to be a reasonable amount of time to ask of the user in order to characterise their land parcel and/or plan their garden. The DSS can be stopped (and the work saved) at virtually any time in the process of carrying out this DSS (except during the running of the Garden Layout plugin/method).

5.3 Recommendations for Future Work

The following paragraphs suggest changes that can be made to the DSS prototype that will serve to increase its usability and functionality.

The recommended amendments are ones that seem feasible and could be a benefit to the user due for increased ease of use or increased capabilities of the DSS. The inclusion of a time element to this DSS prototype, where the user could see the progress of the growth of the plants over time, may be helpful in determining the life cycle of each plant. This could be done in SketchUp with a vertical representation by using the projected plant height to show the shadows cast and to illustrate when the next succession of a particular vegetable should be sown when its growing cycle is over and less than that of the growing season dictated by climate. The potential shading by a plant onto other plants could also be a variable within the DSS to assist with the development of a better garden layout. Also related to the time element over a growing season is the ability to link the same garden plot over multiple growing seasons to take advantage of crop

rotation to avoid depleting areas of the soil of nutrients by potentially planting the same vegetables in the same location year after year.

Because the urban landscape contains trees (fruiting and non-fruiting), many users may have one or more trees on or near their land parcel. Incorporating companion and antagonist relationships between the vegetables in the DSS and surrounding trees as well as information on interactions with root systems may prove valuable to the user. The inclusion of average yields to give the user an idea of how much food they could grow also may be valuable to the user. Using the knowledge of yields the user could potentially input the distribution of vegetables in the garden using garden area percentage or yield values.

A final modification that could greatly increase the ease of use of this DSS prototype is the addition of a personalized document which would include the garden name (to differentiate between other gardens), inputs (reaching length, site conditions, etc.), and tabular information currently located in VegetableDatabase.xls that would only show the vegetables that appear in the planned garden. This would assist the user in identifying the vegetables and would save the user time if she wanted to make an easily accessible reference document herself. These recommendations are from the perspective of the researcher, and as this DSS attracts users, these users may give feedback in the form of comments and suggestions for the purpose of improving its functionality and ease of use.

The use of this DSS prototype was originally thought to be a 'building block' of a larger DSS that would incorporate individual food gardens on a larger scale. It is proposed that future work incorporating this DSS be related to food networks among the urban population. It is foreseeable that urban residents using this DSS may have excess vegetables after harvest and creating networks of food producers on different scales (block, neighbourhood, ward, etc) may connect people willing to buy or barter produce grown locally. Gardens within a region could be planned to accommodate collective food preferences and quantities and take advantage of varied site conditions and sizes of parcels. This networking may also incorporate expertise of food producers; certain

residents may have experience growing particular vegetables, and provided their site conditions coincide with the vegetable(s) chosen to grow, that resident may be responsible for producing only a few types of vegetables. In that same vein, networks not only of food, but of knowledge, experience, and supplies may coexist to provide support among these local food producers and urban residents.

Chapter 6 References

- Baker, L.E. (2004). Tending Cultural Landscapes and Food Citizenship in Toronto's Community Gardens. *The Geographical Review* 94(3) 305-325.
- Bell G.M., Danneberger T.K, McMahon M.J. (2001). Spectral Irradiance Available for Turfgrass Growth in Sun and Shade. *Crop Science*. 40 189-195.
- Beck, A. (2008). *The Canadian Edible Garden, Vegetables Herbs, Fruits, & Seeds*. Edmonton, AB Canada. Lone Pine Publishing.
- Beck, T. B., Quigley, M. F., Martin, J. F. (2003). Emergy Evaluation of Food Production in Urban Residential Landscapes. *Urban Ecosystems*, 5, 187-207.
- Bennett, J. (1996). *The New Northern Gardener*. Willowdale, ON Canada. Firefly Books Ltd.
- Bird R. (1990). *Companion Planting*. Toronto, Ontario, Canada. Sterling Co. Inc.
- Bomford, M. K. (2009a). Do Tomatoes Love Basil But Hate Brussels Sprouts? Competition and Land-Use Efficiency of Popularly Recommended and Discouraged Crop Mixtures in Biointensive Agriculture Systems. *Journal of Sustainable Agriculture* 33(4) 396-417.
- Bomford, M. K. (2009b). *Companion plant spacing calculator*. Retrieved March 19, 2010 from <http://organic.kysu.edu/CompanionSpacing.shtml>
- Djiang, J. (August 31, 2009). *The White House Blog: The Story of the White House Garden*. Retrieved on February 9, 2010 from <http://www.whitehouse.gov/blog/The-Story-of-the-White-House-Garden>
- Fitzgerald, M. (2007). *Learning Ruby*. Sebastopol, CA, United States of America. O'Reilly.
- Fulton, H. (2002). *The Ruby Way*. Indianapolis, IN, United States of America. Sams Publishing.

- Gardener's Supply Company (n.d.). *Kitchen Garden Planner*. Retrieved January 14, 2010 from http://www.gardeners.com/Kitchen-Garden-Planner/kgp_home,default,pg.html
- Google (2009a). *Google SketchUp Ruby API*. Retrieved December 12, 2009 from <http://code.google.com/apis/sketchup/>
- Google (2009b). *SketchUcation Community Forums*. Retrieved December 13, 2009 from <http://forums.sketchucation.com/viewforum.php?f=180>
- Google (2010a). *Download Google SketchUp 7*. Retrieved March 8, 2010 from <http://sketchup.google.com/download/gsu.html>
- Google (2010b). *Download Google Earth for PC, Mac or Linux*. Retrieved March 8, 2010 from <http://earth.google.com/download-earth.html>
- Growing Interactive (2010). *GrowVeg.com*. Retrieved January 14, 2010 from www.Growveg.com
- Hill, S.B. (1975). *Ecological Agriculture Projects: Companion Plants*. Retrieved December 16, 2009 from <http://eap.mcgill.ca/publications/EAP55.htm>
- Hopp, S. L., Gussow, J.D. (2009). Comment on "Food-Miles and the Relative Climate Impacts of Food Choices in the United States". *Environmental Science & Technology*, 43(10) 3982-3983.
- Hough, M. (1995). *Cities and Natural Process*. Routledge, (pp. 207-215). New York, NY, United States of America. Routledge.
- Howe, J., Wheeler, P. (1999). Urban Food Growing: The Experience of Two UK Cities. *Sustainable Development*, 7, 13-24.
- I-Farm Team (2010). *I-farm tools.org*. Retrieved March 16, 2010 from <http://i-farmtools.org/>
- Jeavons, J. (1995). *How to Grow More Vegetables: than you ever thought possible on less land than you can imagine*. Berkeley, CA United States of America. Ten Speed Press.

Jolliffe, P.A. (1997). Are Mixed Populations of Plant Species More Productive Than Pure Stands? *Oikos*, 80(3) 595-602.

Kuepper, G., Dodson, M. (2001). *Companion Planting: Basic Concept and Resources*. Retrieved December 16 2009 from <http://attra.ncat.org/attra-pub/complant.html>

Mendes, W., Balmer, K., Kaethler, T., Rhoads, A. (2008). Using Land Inventories to Plan for Urban Agriculture. *Journal of the American Planning Association*. 74(4) 435-449.

Niemiera, A.X. (2009). *Intensive Gardening Methods*. Virginia Cooperative Extension. Retrieved December 16, 2009 from <http://pubs.ext.vt.edu/426/426-335/426-335.html>

Plangarden (2010). *Plangarden.com*. Retrieved January 14, 2010 from www.plangarden.com

Region of Waterloo Public Health. (November, 2005). Food Miles: Environmental Implications of Food Imports to Waterloo Region. Retrieved December 09, 2009 from [chd.region.waterloo.on.ca/web/health.nsf/0/F9E487C67FAC45E885256FE90060ADF6/\\$file/Food_Miles_Report.pdf?openelement](http://chd.region.waterloo.on.ca/web/health.nsf/0/F9E487C67FAC45E885256FE90060ADF6/$file/Food_Miles_Report.pdf?openelement)

Riotte, L. (1998). *Carrots Love Tomatoes: secrets of companion planting for successful gardening*. Pownal, Vermont, USA. Storey Communications Inc.

Stonehouse, P.D. (2004). Sustainability Issues in the Agri-Food Sector in Ontario, Canada. *Journal of Sustainable Agriculture*, 23(3) 109-124.

The GIMP Documentation Team (2009). *GNU Image Manipulation Program: User Manual*. Retrieved December 13, 2009 from <http://docs.gimp.org/2.6/en/>

The GIMP Team (2009). *Downloads*. Retrieved March 8, 2010 from <http://www.gimp.org/downloads/>

Weber, C.L., Matthews, H.S. (2008). Food-Miles and the Relative Climate Impacts of Food Choices in the United States. *Environmental Science & Technology*, 42(10) 3508-3513.

Weber, C.L., Matthews, H.S. (2009). Response to Comment on "Food-Miles and the Relative Climate Impacts of Food Choices in the United States". *Environmental Science & Technology*, 43(10) 3984.

Appendices

Appendix 1 User Instructions

A1.1 Introduction

This set of instructions aims to assist in the use of the decision support system Urban Food Production: A Prototype Decision Support System. This DSS includes the plugins Shadow Analysis and Garden Layout.

Before following these user instructions, some additional actions on the part of the user are required (special note: only computers that use Windows XP or more recent can use this DSS):

1. **Download Google SketchUp and Google Earth.** These applications are necessary to provide a foundation for the decision support system.
 - Copy <http://sketchup.google.com/product/> into your web browser.
 - In the left hand menu you will find both SketchUp 7 and Google Earth listed. Click on these menu items and follow the downloading instructions (choose the 'engineering' style as the SketchUp template). Make note to which directory these applications are saved.
2. **Download GIMP.** If you already have an image manipulation program such as Photoshop which offers Multiply and Transparency functions, allows multiple images to be opened as layers, and the user is familiar with the alternate program, downloading GIMP may not be necessary.
 - Copy <http://www.gimp.org/downloads/> into your web browser and choose the GIMP installer for Windows.
 - Make note to which directory GIMP is saved.
3. **Make sure to have a spreadsheet viewer.** A vegetable 'database' is provided in a spreadsheet, written in Microsoft Excel, and saved as an .xls file.
4. **Copy the TotalPackage.rb plugin into SketchUp.** Navigate to the Google folder > Google SketchUp 7 > Plugins. Save TotalPackage.rb directly into Plugins.
5. **You should be ready to start!**

A1.2 List of Useful Tools

Below is a list of tools in SketchUp that you may need to refer to during your use of this decision support system. Shortcuts and brief descriptions are included.

Camera: press Alt C, then P, then T (for top) and F (for front). Top view shows the x-y plane from the perspective of the positive z-axis. Front view shows the model from the view-point of the x-y plane (the ground plane is not visible)

Move: press M. The cursor will turn to a cross with arrows at each end (this is also the tool's icon in the toolbar). For rectangles: clicking on the face of the rectangle and moving the cursor will serve to move the rectangle in the same direction. Selecting an edge (making sure the face is not selected) will allow the user to move the edge independent of the other edges. For components: the move tool can also rotate the object. When the cursor hovers over the component small red crosses appear, if these are clicked on the function is now like that of the Rotate tool.

Orbit: press O. Use the left mouse button and move the cursor to orbit around your model in SketchUp. If you are looking to view directly over top of your model or in the line-of-sight of the ground plane, use the camera views.

Pan: press H. The cursor will turn into an image of a hand (this is also the tool's icon). By clicking the viewing area in SketchUp and moving the cursor the view will change but the direction of perspective will not change.

Rectangle: press R. The cursor will turn into a rectangle (this is also the tool's icon). Click a point to initiate the drawing of the rectangle. The dimensions of the rectangle will appear in the bottom right corner of SketchUp. This first point will be a corner of the rectangle and clicking again will create the corner diagonal to the original point.

Rotate: press Q. The cursor will turn into a protractor (this is also the tool's icon). To rotate a rectangle (for components see Move) in the x-y plane, make sure the cursor is blue and click on the rectangle's face and click another point to define the line of zero degrees. Move your cursor to rotate and click the cursor when you are satisfied with the rectangle's position.

Select: press space bar. The cursor will remain an arrow (this is also the tool's icon). Clicking the face of a rectangle will select and shade the face (not including the edges); double-clicking the rectangle will select the face including all edges (the face will be shaded and the edges will turn blue); clicking on an edge will select that particular edge and turn it blue; for components, clicking the component will serve to select it and an edged box around the component will appear in blue.

Tape Measure: press T. The cursor will turn into a tape measure (this is also the tool's icon). Click on two points to measure the distance between them. The length of the line segment will show up in a text box near the cursor as well as in the bottom right corner of SketchUp.

Zoom: press Z. The cursor will turn to a magnifying glass (this is also the tool's icon). Clicking a point within SketchUp and moving the cursor forward and back will zoom in and out, respectively.

Zoom Full Extents: press Ctrl-Shift-E. The tool's icon is a magnifying glass with four arrows. Once the icon or command is pressed, SketchUp will zoom in or out to the full extents of the model using the current perspective.

A1.3 Shadow Analysis

In this section of the decision support system you will use Google Earth, Google SketchUp and GIMP (or similar image manipulation program) to make a 3-D model of the relevant (shadow casting) objects on or near the property to contain the garden.

Completing the entire Shadow Analysis step is not required to start the Garden Layout section; if you have sufficient information of the sun conditions (knowledge of areas of full sun, partial sun/shade, and/or light shade) of the area, you may complete **step 1 and 2**, and proceed to Garden Layout.

1. **Open Google Earth and Google SketchUp.** Make sure both of these applications are active before starting. If you do not have these programs, see the introduction of the User Instructions.

2. **Import a Google Earth Image to Google SketchUp.** Find your location using Google Earth and import that image to SketchUp.

- In Google Earth, type in the address of the parcel you wish to evaluate and zoom in to the parcel making sure to include significant shadow-casting objects such as nearby buildings, trees, etc.
 - Double-clicking on a point will zoom in and single-clicking will cease zooming.
 - The +/- slider bar on the top right of the window will zoom in or out. Using the + has the unwanted effect of rotating the view of the x-y plane: use the top-most tool's down/South arrow to make the view overhead.
- Once the image is close enough that it encapsulates the appropriate features, open the Google SketchUp browser, and either:
 - Navigate to the Tools menu in the main toolbar, then to the Google Earth submenu and then click on Get Current View within that menu, or;
 - Navigate to the icon of a blue and white sphere with a yellow arrow pointing down. If you hover over it, a text box containing "Get Current View" will appear. Click this icon.
- The 2-D Google Earth image will appear in the x-y plane of SketchUp.

3. **Make a 3-D model of the relevant features.** Using the Google Earth image as a template and the components in the Plugins>Shade Tools menu you can make a model of the parcel in question. Use the Move tool to move and rotate these objects. Note: all of the components will appear at the origin. To prevent overlap of components, make components of objects that are farthest away from the origin first.

- Fence: This fence looks like it is made with 2"x4"s. The 'boards' are four inches wide and one inch apart. If you have a chain-link fence you might want to use the line tool to draw where the fence lies, but because chain-link fences

don't cast a significant shadow, a fence may not be needed. The fence will be laid along the red (x) axis.

- House: This 'house' is a rectangular prism with a triangular prism as the roof. This component may also be used for a shed or similar structure. You are asked to provide the height to the house to the top of the peak, the height to the beginning of the roof, the length (along the side where the roof touches the wall), and the width (along the side of the peak). The overhang is also asked for. If you measure the width of your house from SketchUp, make sure that twice the overhang plus the width of the house is equal to that measurement.
 - Warehouse/Apartment Building: This plugin should be used to make a structure that is a rectangular prism of any size. An input box asks for the structure's height, width, and length.
 - Coniferous (Pine) Tree: An input box will ask for trunk height (the part that is exposed) and diameter, as well as top and bottom canopy widths. We assume that the trees are rotationally symmetric about their trunks. A 2-D component will be produced, place it in the center where the tree trunk should be.
 - Deciduous (Leafy) Tree: An input box will ask for the diameter and height (the exposed section) of the trunk and the height/width of the canopy. The canopy is the leafy 'ball' of the tree; it will show up as a circle. This component is also 2-D but assuming cylindrical symmetry, the shadows will be the same as a spherical canopy. Place this component at the center of the tree trunk on the Google Earth image.
 - Hedge: An input box will ask for the height and width of the shrub, this plugin is intended to be similar to a hedge. The component produced is a 2-D rectangle that casts shadows as if it were cylindrically symmetric about its z-axis.
4. **Make sure everything is sitting on the ground.** Use the Camera (Alt C, then S, then F) to get a view of the side of your model. You can also use Orbit to change the orientation. This step will ensure proper shading.

- To move an object up or down, use the Camera>Standard View> Front position, select the item with the move tool and move the cursor straight up and down.
5. **Save your work.** You will need this for Garden Layout plugin. Save this .skp file as something descriptive so it is easy to recognize later (MyAddressModel.skp).
 6. **Select the Google Earth image with the Select tool.** This image will have a red outline when it is selected. Make sure it is the only thing selected. Note: the view of the model is not important; the Shadow Analysis will orient and zoom automatically.
 7. **Apply the Shadow Analysis plugin.** Navigate to Plugins>Garden Tools>Shadow Analysis once your model is ready and the Google Earth image is selected. Note: if the Google Earth image was not selected you will have to revert back to the previously saved version.
 - Pick a garden name and enter it in the input box. Keep in mind that this will be part of the file name of each image, so don't make it too many characters.
 - Year, start/end month, day, and time zone are asked for in an input box. The start/end date should be the last/first frost date (in southern Ontario the last frost is near the end of May).
 - Saving the images: it is recommended that you make a folder in your C:\ directory in which to deposit the images to be produced. Leave the input box blank if you prefer to save all images directly to the C:\ directory.
 8. **Use GIMP to manipulate your shadow images.** Open GIMP and navigate to File > Open as Layers. Navigate to the folder where your Shadow Analysis images are saved. Select all images that were produced from this SketchUp.
 9. **Adjust Opacity and use the Multiply mode.** The images will appear in the main GIMP window as well as the Layers toolbar.
 - Select the "[Garden Name]Background.jpg" in the Layers toolbox and move it to the last position or the second last if a white 'Background' is the last item.

- Change the Opacity to 10 or 30 percent and the Mode to Multiply for each image except for the “[Garden Name]Background.jpg” image at the end of the list.

10. Use this image to find the sun conditions of your property.

- To get a more general sense of whether it is sunny or shady on the possible garden location, observe the shadow patterns from all images multiplied together.
- To get a sense of how many hours a day a part of land is shaded for, look at individual months by clicking on each eye icon next to the images of all other months. Remember to keep the “[Garden Name]Background.jpg” image visible. While looking at the multiplied images of a single month, each shadow represents an hour. For example, if there are three shadows overlapping an area, and eight images in that month, then that area has at least 5 hours a day of no shadow (or full sunlight). This will be important when choosing vegetables in the next section, Garden Layout.

11. Save your layered image(s). Choose Export when asked to merge layers, then Save.

A1.4 Garden Layout

This plugin is designed to help the user to plan a garden on a parcel of land. It is required that the user is familiar with the conditions of their land for optimal planning. If you have not completed the Shadow Analysis make sure to complete its first two steps. This section will require the use of SketchUp and reference will need to be given to the Microsoft Excel spreadsheet to reference vegetables used.

1. **Draw a rectangle.** This rectangle will represent your garden. You may want to use the Tape Measure tool in the proposed location of your garden in order to know what dimensions you would like for your rectangle.
 - Activate the rectangle tool.

- Click the x-y plane and drag until the desired dimensions are reached. The dimensions will be shown in the bottom-right corner of the SketchUp window. Don't worry if you are unsure of the dimensions or want to change later, the Move tool can be used to change its length or width.
 - Rectangle not solid? Because the Google Earth image and the rectangle share the ground plane, your rectangle's face may look like shattered glass. This is normal and will not disrupt the functioning of the program.
- 2. **Position the rectangle.** Use the Move and Rotate tools to put the rectangle in your preferred garden location. Note: make sure to leave enough room around the garden in order to access it from all sides (i.e., at least a path-width space between the garden and any structures like fences, sheds, etc.).
 - Activate the Move tool and click on the face of the rectangle. Move the rectangle to the appropriate location.
 - If you need to rotate the rectangle, activate the Rotate tool. Click on two points (one on the face of the rectangle) that will define your zero-degree line. Make sure the protractor is blue when you make this line. Move the cursor around in small, controlled motions until you are satisfied with the rotation of your rectangle, then click at that location.
 - If you need to change the length or width of the rectangle use the Move tool to select an edge (to change the width, choose the lengthwise side), and move it in the appropriate direction.
 - Having trouble? Make sure the rectangle's face is not selected (if the rectangle is covered in dots/shaded it is selected) by hovering the Move tool over the rectangle and off. If the rectangle stays shaded use the Selection tool to deselect the rectangle by clicking on the 'ground'.
 - Use the above steps again until the rectangle is positioned and sized to your liking.

3. **Make sure your rectangle is lying flat on the ground plane.** Check that the rectangle is flush with the x-y plane.
 - Use the Camera (Alt C, then S, then F) and Orbit tool see the view parallel to the ground.
 - If the rectangle is not flush, use the Move tool to select the rectangle and move the cursor directly downward with no side motion.
4. **Save your work.** This will allow you to:
 - come back to an already-positioned garden in case you decide stop now and resume later.
 - make variations on the vegetables going into the garden. It will save you going through this first two steps multiple times.
 - make variations of the location of your garden.
 - have a backup just in case!
5. **Select 3 items:** the face of the rectangle, leftmost edge, and bottom edge. This step is necessary in order to use the Layout plugin.
 - The 'bottom edge' or 'foot' of the garden will be the edge of entry for the rows. The rows will run perpendicular to the foot of the garden. The leftmost edge is taken from the perspective of the foot of the garden.
 - Activate the Select tool and click on the face of the rectangle. The rectangle is selected when shaded.
 - With the Ctrl-key depressed click on the two edges that will represent the foot of the garden and the leftmost edge. These edges will share a corner and should turn blue when selected.
 - The face and these two edges should be the only items selected. If the shift key is depressed and you click on an edge but it is not selected, start again.
6. **Start the Garden Layout plugin.**
 - Navigate to the Plugins menu, then the Garden Tools sub-menu. You will then click on the Garden Layout plugin within this folder.
 - Error Message?

- If you get an error that states that an incorrect number of items have been selected, start again from step 4. Make sure the rectangle is the only thing selected when you start.
- If you receive an error message stating that there are not two edges and a face selected, start again from step 4.

7. **Choose your units.** Imperial (inches) or metric (centimetres).

8. **Choose your reaching length and path width.**

- The maximum reaching length is the greatest distance that would be comfortable for the user of the garden to hold for extended periods of time (for planting, weeding, and other maintenance).
 - Note: Make sure the length of your garden is at least twice the length of your maximum reaching distance. If the length (defined by the left-most edge of the garden) of your garden is less than two maximum reaching lengths, please enter half that length as your reaching length. If this step is not taken, an insufficient layout will result.
- The path width is the width that will be most appropriate for the uses of the user. Some things you may want to consider: will a wheelbarrow be used on a regular basis? How large is the garden? How much room is needed to comfortably weed or perform maintenance? Who will be using the garden?

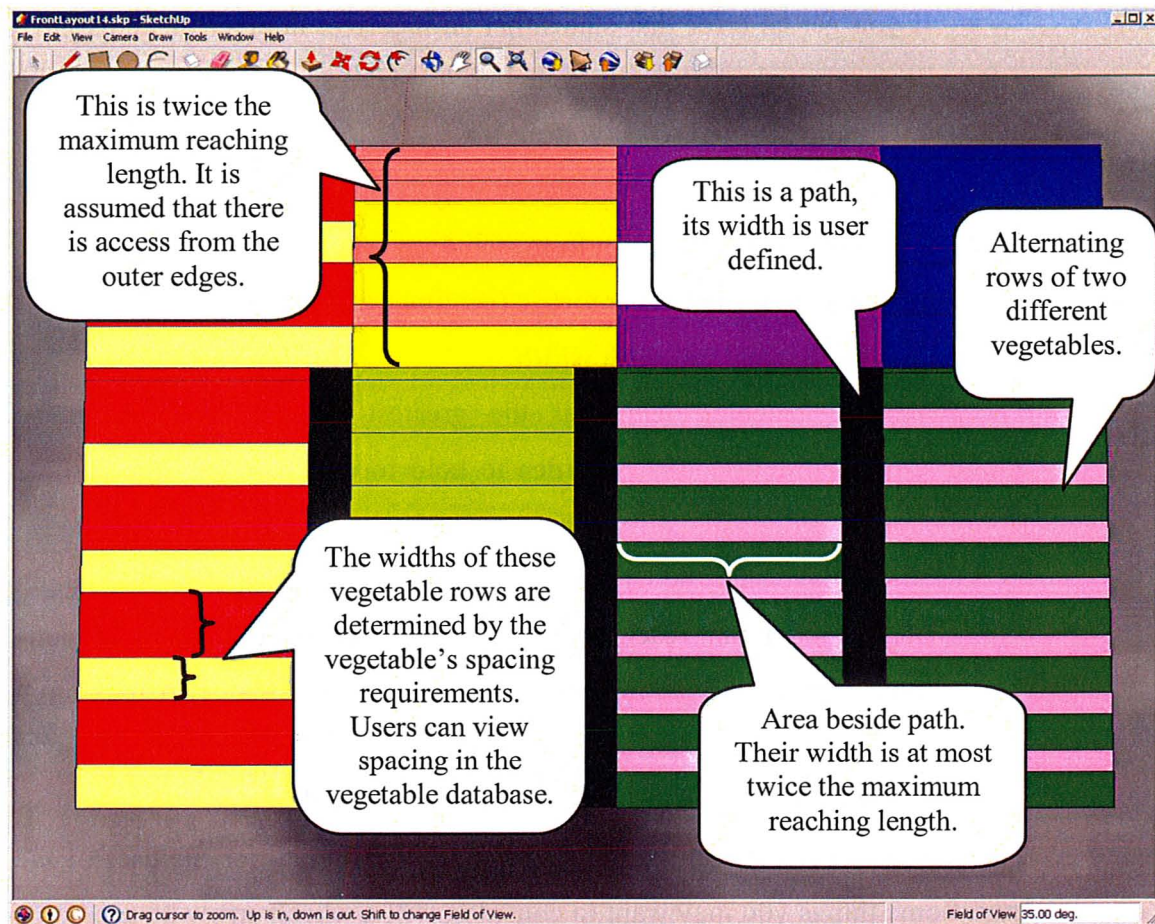


Figure A1.1 A representative garden layout output. Its purpose is to show how the user-input values correspond to dimensions in the layout.

9. **Choose the conditions of your garden.** This set of drop-down menus allows you to set the criteria for the vegetable list to follow. 'All' is the default in each case; it means all choices for that category.

- Decorative Edibles: Options are All or Yes. Decorative edibles are edible plants (vegetables herbs and flowers) that are also attractive. A decorative edible garden could replace a landscaped garden.
 - All: all edible vegetables will be included, not just decorative/attractive ones.
 - Yes means that only decorative/attractive edible plants will appear in the vegetable list.

- Sun Conditions: Options are All, Full-Sun, Partial-Shade, and Light-Shade.
 - All: plants with any sun requirement will be included.
 - Full-Sun: plants that grow well with more than 6 hours of sun per day.
 - Partial-Shade: plants that receive full sun for part of the day (4-6 hours).
 - Light-Shade: plants that are shaded for most of the day, but receive some direct light. A garden shaded by a tree, but some direct sunlight still filters through the leaves, is an example of Light-Shade.
- Perennial/Biennial or Annual: Options are All, Perennial/Biennial, and Annual.
 - All: plants with any type of growth period will be included.
 - Perennial/Biennial: plants that require three/two years to complete their life cycle.
 - Annual: plants that germinate, produce, and go to seed in one growing season.
- Sowing Conditions: Options are All, Indoors, and Outdoors.
 - All: plants with any sowing requirement are included.
 - Indoors: plants that can be germinated indoors.
 - Outdoors: plants that can be germinated outdoors.
- Error Message? If your particular combination of criteria produces a list containing zero vegetables, you will be asked to re-evaluate your selection and choose another set of criteria. Think about the criteria that are most important for your particular situation.

10. Choose your edible vegetables, flowers, or herbs. Looking at the Vegetable Information spreadsheet may be helpful before picking vegetables.

- Choose from the list of vegetables in the drop-down menu that has been selected to fit your garden criteria.

- Use the up and down arrow keys or the first letter of the name of the vegetable you would like to scroll through that letter category. Press 'Enter' to select that plant.
- Or use the cursor to select the drop down menu and click on the vegetable to choose it.
- After each vegetable selection a message box will appear listing the plants that have been selected and asking if you would like to choose more vegetables. Press 'Enter' to choose 'Yes'.
 - If you accidentally choose 'Yes' when you wanted to choose 'No', just choose a vegetable that you have already chosen and when you reach the 'More Vegetables?' message box again, choose 'No'.

11. Check out your new garden layout!

- Using the Vegetable Information spreadsheet provided, match the colour in the layout to the vegetable colour in the spreadsheet to use your layout.

Appendix 2 Hypothetical Trial Layouts

A2.1 Backyard Trials

For all graphical layouts of the backyard trials, see the electronic appendix.

A2.2 Front Yard Trials

For all graphical layouts of the front yard trials, see the electronic appendix.

Appendix 3 TotalPackage.rb Code

For the Ruby code, TotalPackage.rb, see the electronic appendix.

Appendix 4 Vegetable Database

For a spreadsheet (VegetableDatabase.xls) and printable (VegetableDatabase.pdf) versions of the vegetable database for use with the prototype DSS, see the electronic appendix.

