Importance of Feature in Indoor Mapping and Doorways Detection with Depth-Sensing Camera

Submitted May 2015, in partial fulfilment of the conditions of the award of the degree BSc Honours Computer Science.

Adam Weisen Goh awg04u

Supervisor: Dr Andrew French

School of Computer Science and Information Technology

University of Nottingham

I hereby declare that this dissertation is all my own work, except as indicated in the text:

Signature:

Adam Weisen Goh

Date: 08/05/2015

ABSTRACT

In recent years the advancement in robotics had improved solutions to Simultaneous Localization and Mapping (SLAM) problems. Feature-based SLAM then gained tractions as it uses cost effective sensors such as cameras to capture visual features. However, the presence of doorways and the failure to detect them results in erroneous performance of feature-based SLAM. Door detection then became one of the more prominent researches in the area of robotics. In this project, basic door properties such as door indentation and change of colour in door are explored in determining the accuracy of door detection. First, depth and colour data are collected along a hallway that contains a doorway. The data frames are then stitched together to form a panoramic image. The stitched depth and colour image will then be used to detect for change of depth and change of colour, and classifying the existence of a door. The results of the project show mixed results. The stitching of images is successful regardless of each frame's exposures. Successful door candidate was extracted when a difference in colour and in depth was found. The system failed, however, to extract door candidate when only a difference in depth was found. The system is evaluated to be unsatisfactory and can be improved with future works and better algorithm designs.

ACKNOWLEDGEMENT

This project wouldn't be completed without the help of others:

Gratefulness and huge appreciation to my supervisor, Dr. Andrew French, for being supportive, generous in guidance and time, and utmost patience with me.

Thank you to the school of Computer Science for allowing the collection of data within the building and providing the funds in purchase of sensors used in this project.

Thank you to all family members and friends who supported me throughout the completion of this project.

ABBREVIATIONS

Below is a list of abbreviations that will be used throughout the course of the document:

RGB : Red Green Blue

TOF : Time of Flight

SLAM : Simultaneous Localization And Mapping

GUI : Graphic User Interface

QQVGA : Quarter-Quarter Video Graphics Array

VGA : Video Graphics Array

RGB-D : Red Green Blue - Depth

OCR : Optical Character Recognition

IDE : Integrated Development Environment

RAM : Random Access Memory

RANSAC : Random Sample Consensus

TABLE OF CONTENT

TABLE OF FIGURES

1 INTRODUCTION

Common SLAM research and projects often span over a few years of research and tends to be difficult while requiring high attention to details and precise choice of algorithms to be efficient. When developing SLAM systems for an indoor environment, one of the more commonly faced challenge faced is having autonomous robots to generate a representation of the environment environment. Common environment representation includes topological-based(Gilliéron and Merminod 2003), graph-based(Thrun and Montemerlo 2006), line-based(Garulli, Giannitrapani et al. 2005), or occupancy grid(Moravec 1988) maps. This action requires the system to correctly identify doorways to enter or leave rooms, areas, or even elevators.

A door detection problem is a more focused problem in the research area of SLAM. Door detection modules prove to be vital for automated robots to navigate the indoor environment as it indicates the transition between spatial regions in its constructed map. The core challenges in solving a door detection problem have large overlaps with common challenges for indoor SLAM. For example, to locate a door along the hallway, the hallway lines must be found to narrow down the search. To achieve this, a system must be able to overcome the sensor aliasing, where hallways have many linear features with similar contrast that are in parallel or close to the hallway lines, making high precision rates difficult. (Shi and Samarabandu 2006).

Depending on the need of the system, a door detection module may also vary its specifications. For example, a SLAM system that requires navigating through doorways will need information such as width of the doorway, its orientation and detected position of the doorway(Zender, Mozos et al. 2008). Meanwhile a stationary security camera may just need to detect accurately the doors in a single hallway and the motion of doors opening within that hallway. Furthermore, with indoor buildings varying their architecture designs with different spaces, shapes and lighting, it is difficult to produce an all-rounded door detection module that work in every kind of hallways that prove valuable to all indoor system.

To narrow down the focus of the project's problem statement, the project intends to investigate the available features and properties that can be used to identify a doorway. Ideally, the system aims to successfully map a partial hallway with the ability to detect doorways along it. The door detection module makes several assumptions to reduce environment variables. Firstly, the system is limited to features found along the hallway wall. The sensor is positioned perpendicular to one side of the hallway, and captures data along the hallway in a parallel motion. Note that this method of capturing data means several sub-problems of door detection are irrelevant, such as searching for hallway lines and estimating the width of a hallway. The result of the door detection module is an extracted door candidate from a stitched panoramic image of the hallway.

With the given problem statement, the choices of sensor proves vital in effectively capture necessary data. Several issues were considered before concluding

having a combination of both color and depth sensors, which Time-of-Flight cameras are able to do, to capture data. Because hallway lines are not visible in the captured data, a solution must be introduced to tackle the sensor aliasing problem. Depth data proves important in this case as it is able to conclude if the data captured belong to the same hallway by attempting to fit a model into it. The geometry of the door is also partially captured and having door detection algorithm solely based on it will not be suitable. Changes in color data can be exploited to detect a possible doorway. Furthermore, There's a need to capture not just any features that a door may have, but also able to consistently detect the hallway and differentiate it from a doorway. These issues will be tackled in this project with the use of both color and depth data.

1.1 Aims and Objectives

The aim of the project is to study the importance of features along a hallway in detecting doorways and wall along a hallway utilizing a short distance depth-sensing RGB camera. The project will explore techniques and approaches of door detection using difference in depth and difference of colour in data. Image registration techniques used to create an accurate panoramic image from the given frames is also a part of the implementation. Finally, the system should also be able to describe hallway walls with linear regression to find the best line models, extracting doorways given line models and detecting presence of doors from a change of colour in colour data.

The **objectives** of the project are to present the following deliverables:

1) A command-line user interface to specify the dataset folder used in the system.
2) Create "Panoramic" image by image stitching algorithms on depth and color data.
3) Hallways wall detection by devising a best fitting models with the given depth data
4) The system is able to detect door candidate using both color and depth data.
5) Collect any visual information along the hallway such as door signs, warnings texts, or instructions.
6) Testing of the system for different types of situations and discusses the result of the tests.

2 LITERATURE REVIEW

This section discusses the research area in a general and high level point of view in respect to the methodology and context view of the specific problem. First, the mapping problem in SLAM is explained and how it relates to the importance of door detection as a feature of an environment. Then, door detection is explored in context of the indoor environment. The choice of sensor used in door detection techniques is detailed. The context of the problem and challenges faced are also described. Finally, the context of various technical terms and algorithms are described as self-help background knowledge.

2.1 Indoor Navigation and the importance of feature detection

Simultaneous Localisation and Mapping (SLAM), usually deemed as a chicken-or-egg problem, involves updating an agent's current belief state in a map while constructing the map from the environment through various sensors given the agent's believed state. Throughout the years SLAM problems had been tackled by approximating solutions such as filtering algorithms like particle filters and Kalman filters, and SLAM systems were recently brought to the limelight with consumer robot vacuum cleaners or self-driving cars in the consumer market.

Solutions to SLAM problems often require extracting abstract features from given percepts. These features are useful for localisation purposes by marking them as landmarks to approximately locate the agent's position in the environment(Sim and Little 2006),(Gil, Reinoso et al. 2010). They can be unique edges, corners, lines, or curves in the environment. These features also belong to a core part of the environment construction process.

The mapping problem within SLAM systems can be approached from various directions. Recent work diverted from laser range data with probabilistic pose and landmark estimations, to solely relying on visual image to solve SLAM problems. These works also rely heavily on unique features in the environment(Migliore, Rigamonti et al.),(Sim and Little 2006),(Gil, Reinoso et al. 2010). An example of such work is VisualSLAM(Gil, Reinoso et al. 2010). VisualSLAM require robust feature matching that is invariant to scale and perspective transformation for every camera pose estimation. In the context of an indoor environment, these systems generally faces sensor aliasing problem such as having similar geometry shapes and structures without significant features.(Siegwart and Nourbakhsh 2004) Choices of sensor must also take into account the environment the robot is in, if the sensors provide enough details to generate meaningful features with adequate accuracy.

Another advantage of feature based SLAM is reducing the computational requirements and complexity of the map generated. As feature-based SLAM only deal with features extracted from color data, there are less data that is required to be processed to extract meaningful features. For example, if the detected features are edges, lines, and corners taken from a monocular camera, the computation done by the algorithm to generate a pose estimation of the robot and mapping of the environment may be simpler than a 3-dimensional point cloud map generated from a laser range finder.

The importance of features detected in environment construction is closely related to the type of sensors used for data collection. Kim and Nevatia(1998) pointed out how having the right choice of sensor is just as important as understanding the complexity of the detected object during a recognition task. Therefore, the suitable choice of sensors and detected abstract features are vital in intelligent agents' decision making within a SLAM system, especially in defining the reliability, validity and safety of the system.

Zender et al.(2008) described an approach on creating conceptual representations that clearly illustrate such a relationship. Different types of features and representations were utilized differently for different purposes. A primitive, line-based metric map was first generated with laser range finder. The map will be used for robot's localization. Then, shape features such as edges, corners and lines are used for generating a navigation graph. A navigation graph represents the context of "free-space" and "connectivity" by the notion of a node. Node classifies an area of enclosed space to be a room, hallway. A node may also be in nature of transitional, such as a doorway, which translate the position of the robot from a space to another. The paper then followed up on the importance and need to detect doorway nodes, mentioning how shape features constitute doorway nodes, which "indicate the transition between different spatial regions".

2.2 Door Detection and its importance in indoor navigation

In relation to the previously mentioned paper, Kim and Nevatia(1998) also described a novel mapping solution called "*s-map*". "*s-map*" represents objects based on their functionalities which are characterised by their significant surfaces, such as a door. For example, a door has consistent and meaningful features such as the door frame and door panels. These features signify an opening passage to another area and have a means of closing the passage. If treating the indoor environment as topological regions, a doorway can be seen as transitional boundary between regions.(Kim and Nevatia 1998) These papers have a common theme towards their opinion on the importance of doorways detection. Being able to detect doorways therefore becomes a crucial factor that must be considered in autonomous robots' navigation component to maneuver through indoor environment.

The practicality of doorway detection techniques also remained relevant in the research area of indoor SLAM at recent times. In Ekvall, Kragic, and Jensfelt(2007)'s work, a service robot was required to detect and deliver objects while navigating around the indoor environment. In aid of its navigation, the robot needed to partition the map in an automated manner by hypothesising the existing of a door. (Ekvall, Kragic et al. 2007)This indicates that having a robust door detection solution aids in the effort of automation and mapping, allowing it to perform other tasks with better precision. Vasudevan, Ga¨chter, Nguyen and Siegwart(2007) also mentioned door detection being a core part of object recognition in indoor environment. The authors described doors detected were "used in the context of place formation".

2.3 Choice of Sensors in Door detection techniques

Over the years door detection techniques implemented had span across many different type of sensors, and several perspectives and angles are tried to obtain the optimum detection rate. Several approaches even considered the importance of identifying hallway lines, either through novel calculation of scan points from laser range finder(Fernández-Caramés, Moreno et al. 2014), or fitting line models through an pre-processed video captured frame image using RANSAC to determine the four hallway lines. (Shi and Samarabandu 2006)

Door detection techniques that uses color data from cameras exploits the structured environment that indoor spaces have, as well as common geometry features that a door possesses. Implementations by Murillo, Koˇseck´a, Guerrero, and Sagu´es(2008) assumes fixed door geometry parameters such as ratio between door width and height as prior knowledge, and computes the likelihood of a door given a prior door model while learning other door parameters (Murillo, Košecká et al. 2008).

Due to sensitivity to light changes and contrasts, implementations that tackle the door detection problem using only color data may face accuracy and reliability issues. There also exist arguments that using only color data for detecting hallway walls are not suitable and possibly unreliable as hallway environments are complex and contain linear features with similar contrast that are parallel to hallway lines(Shi and Samarabandu 2006). These situations paved the way for opting depth data as an additional data source.

Sensors that collect depth data had been commonly used in the past. Laser range finders such as SICK or depth-sensing color cameras (RGB-d) like Microsoft Kinect are common choices of sensor in computer vision research. These sensors are able to detect the distance to an object by illuminating an area with modulated light source and measures the reflected light's flight travel time. Because depth data does not deal with light information, it is invariant to difference of light and ambient light changes, increasing door detection rate. Derry and Argall(2013) chose only depth data for their

implementation due to that specific reason. In their implementation, hallway walls are scanned and projected onto a point cloud to look for potential door opening by finding gaps that fits a hypothetical door width. The found gap represents an open doorway. Door features such as color of the door or texture are not necessary as the location of the door gap and its orientation suffice the system's need as an assistive wheelchair technology.(Derry and Argall 2013)

2.4 Door Detection and recognizing unique door features

With sensor aliasing being a general theme in door detection, it is essential to recognize the constant unique properties that all doorways have. Hensler, Blaich, and Bittel(2010) recognise the uniqueness and importance of door features that differs itself from hallway walls and other items. The implementation by the authors first extracted door candidates by assuming two vertical lines, then seven weak classifiers were then used to detect specific features. The features classified to qualify a candidate as a door are difference of colour from the wall, existence of door knob, door gap, door frame, texture on the bottom of the door, a minimum door width, and door concavity. The results of these classifiers are then combined to a stronger classifier with AdaBoost algorithm.

Recognizing door features for door detection also span across different research that intends to detect doorways in order to resolve problems that it might cause. Lee, Doh, Chung, You and Youm (Lee, Doh et al.) mentioned how the status of a door can vandalise the localization of a robot in the case of a Generalized-Voronoi Graph (GVG). The features used to determine the possibility of a door was a minimum door width, existence of door frames, difference in depth, existence of door knob, and the state of the door. Similar to the Adaboost algorithm approach, these features formed a Bayesian network that calculates the possibility of a doorway in a hallway given the described features.

In a more advance and complex research, Anguelov, Koller, Parker, and Thrun(2004) defined a probabilistic generative model of hallway containing doors as a presentation of the advantages of object-based modelling. The doors as well as walls are parameterized by its visual cues such as shape and color, and its behavior properties such as its motion model. The walls are represented by a straight line segment, while a door object is a segment that can rotate around a hinge, as it is able to capture the orientation of the door as a dynamic parameter. The visual features captured as a door feature are door width and door colour.

From the studied materials it is reasonable to infer that there are certain unique features of doorways that are constant across most indoor environments. Being able to determine and accurately classify these features will highly increase the reliability of door detection modules in detecting doorways. Features that are commonly addressed in the studied materials are door colour, difference in depth, existence of door knob, and a minimum width of a door.

2.5 Problem Specification during Data Collection

Like any other Computer Vision problems, door detection techniques must address the context of the problem specifically. Door detection modules are often mentioned together with autonomous robots or assistive technologies, and optimally functioning in an online manner. (Stoeter, Le Mauff et al. 2000). However, the number of variables exists in the environment complicates the problem, making these techniques difficult to be quick, efficient, and accurate in all situations. The implemented solutions should specify the characteristics of the hallways and doors as indoor environments may vary. Some hallways may be unsuitable for collecting data with specific perspective, while some may have features or architecture that will confuses the module.

By narrowing down the characteristics of the indoor environment being used for data collection and implementation, it greatly reduces the number of variables being addressed and provides much more clarity to the solution. For example, Door detection based on the detection of door frame requires the door frame being in the image(NOZ-SALINAS, Aguirre et al.). This limits the data collected to certain perspectives that perceives the door frame, described to be two vertical lines and a horizontal line being close between them. This limitation in data collection clarifies the practicality of the implementation, eliminating any perspectives that may not hold enough information to aid in the correct classification of doorways.

Specifying the context of the environment also helps in breaking down the problem in better details. A door detection module designed for assistive technology may require data collected to contain the view of door gap to consider if the doorway is wide enough to be passed through(Derry and Argall 2013). If the hallway is too narrow, it will not be possible for a normal pin-hole camera to capture the whole gap in a single frame. This adds another layer of complication to the problem, in which that frames must be stitched together to be a panoramic frame with the full doorway in view. Creating these panoramic images requires implementation of image stitching techniques, involving another large area of computer vision. Derry and Argall(2013) solves this by having panoramic camera and two 2D laser cameras to address this specific issue based on another research(Anguelov, Koller et al. 2004), capturing “shape, color, and motion properties between frames”.

3 Background Knowledge

In this section, several technical terms that will be mentioned throughout the document, but not explained in detail will be introduced and explained. These techniques or terms are not novel and follows a fixed definition. Whenever these terms are mentioned, its referred definition will be as described here.

3.1 Image Stitching

Image Stitching is the term used for algorithms that aligns two or more images to create a single panoramic image. Image stitching is closely related to image registration: finding a transformation model that puts two images into one single coordinate system. Earlier image stitching algorithms attempt to minimize the pixel-to-pixel difference between two images. Some of the solution uses the same mathematical models used in image registration to find translational alignment.

From basic error metrics that minimizes the sum of squared differences to Fourier-based alignment, direct pixel-to-pixel alignment algorithms do not have the ability to differentiate the context of images, where features or textures are unevenly distributed. Furthermore, it is not able to capture the difference in motion models between the two images, where different transformations may occur.

However, advancement in feature-based algorithms led to more feature-based approach in recent image stitching algorithms. Robust and elegant feature detection methods such as Viola and Jones(Viola and Jones 2001)' object detection framework based on Haar-like features, or extracting keypoints based on SIFT algorithm(Lowe (2004)), these algorithms kicks-started feature-based image stitching that overcome some of the fallbacks of direct pixel-based image alignment.

3.1.1Motion Models

Before any image registration or aligning can be done, the motion model must be determined between the images. A motion model describes the mathematical relationship of pixels between two images' coordinate system. Motion model may describe transformations in terms of translational, rotation, scaling, affine, or projective. Each motion model has different characteristics and number of degrees of freedom available. For example. A 2D planar transformation has two degree of freedom, and maintains the scale during transformation, whilst a 3D rotational transformation has three or more degree of freedom, but does not maintain the scale during the transformation.

3.1.2 Translational Alignment

Once the motion model had been determined, the approach towards image alignment can be determined based on the motion model. A translational alignment is the simplest way of image registration to solve translation transformation. Given an template image $I_0(x)$ sampled at discrete pixel location$x_i = \{x, y\}$, translational alignment attempts to find its position in the second image $I_1(x)$ s. Using an error metrics, a pixel-based alignment attempts to minimize the intensity differences of pixels between the two images.

There are different types of error metrics that can be chosen from, depending on the need of the implementation. However, to obtain accurate alignment offsets from the chosen error metrics, one of the most important issues that must be addressed is brightness constancy. Brightness constancy is a perception that corresponding pixel in two different images has the same pixel value, when it in factual that may not hold true. The differences in exposure must be addressed to improve the accuracy of error metrics. Solutions to equalized exposure between two images include normalization of luminance, or assigning gain and weights to pixel values(Szeliski (2010), pg339).

One direct pixel-based alignment technique explored is to directly calculate the **correlation** values between two images. The calculate correlation value shows the correspondence between the pixel values of the first image and the second image. The high the correspondence, the more pixel values are matched between these two images. The objective is to maximize the product of the two aligned images (cross-correlation): (Szeliski 2010)

$$E_{cc}(u) = \sum_i I_0(x_i) I_1(x_i + u)$$

$$where\ x_i = (x_i, y_i)$$

Where I_0 is the first image and I_1 is the second image, x_i = (x_i, y_i) and u is the displacement values u = (u, v).

3.1.3 Normalised Cross-Correlation

Cross-Correlation provides a highly precise and mathematical sound way of pixel-based alignment technique, but it is highly sensitive to changes in illuminance between two images and outliers. If there's a

bright patch in the first image, the maximum cross-correlation value may be shifted to that area(Szeliski 2010). The problem of having different exposure between two images is solved by normalizing them. Normalised Cross-Correlation (NCC) is an error metric that calculates the cross-variance between two images. It eliminates any exposure differences by subtracting each pixel values by its mean values and normalized by the variances of both images. The Normalised Cross-Correlation formula is shown as below:

$$E_{NCC}(u) = \frac{\sum_i [I_0(x_i) - \bar{I}_0][I_1(x_i + u) - \bar{I}_1]}{\sqrt{\sum_i [I_0(x_i) - \bar{I}_0]^2} \sqrt{\sum_i [I_1(x_i + u) - \bar{I}_1]^2}}_{Szeliski}$$

Where

$$\bar{I}_0 = \frac{1}{N} \sum_i I_0(x_i) \qquad \text{And} \qquad \bar{I}_1 = \frac{1}{N} \sum_i I_1(x_i + u)$$

$\bar{I}_0$ and $\bar{I}_1$ are the mean patches of the first image and the second image and N is the number of pixels in the corresponding patch. By dividing with the cross-correlation value with its standard deviations, the exposure between the two images is normalized, hence the term "normalized cross-correlation". The result of Normalised Cross-Correlation always lies in the range of [-1, 1] and will be undefined if either of the image returns a variance of zero, meaning that the pixel values in the patch are all the same(Szeliski 2010).

3.2 Object Recognition

The most common problem for any feature-based algorithms is to detect a desired object reliably. Object recognition techniques usually involve machine learning algorithms, whether being supervised or unsupervised, that aids in recognizing unique and significant features in an image. However, the other approach towards object recognition will be discussed, that is keypoint extraction that identifies features within an image. Keypoint extraction algorithm involves recognizing regions of the image that are have individualistic attributes and can be uniquely identified. Examples of keypoints in an image are edges or corners that have a large gradient of values.

3.2.1Local Invariant Feature Detectors

Tuytelaars and Mikolajczyk(2008) describe a feature as an area in an image with common pattern that differs itself from immediate neighbours. Local features can be described by several properties such as intensity, colour, and texture. One of the more well-known feature detectors is the Harris corner detector, which finds corners in an image by locating areas where a large gradient in all direction occurs. These

corners are used as interest points to identify an object. However, features found with this algorithm are only for a predetermined scale. Hence, Lowe introduced a new feature extraction algorithm that is invariant to the change of scale, the Scale Invariant Feature Transform (SIFT) algorithm.

3.2.2 Scale Invariant Feature Transform (SIFT)

Lowe(2004) describes the SIFT features extracted to be unaffected by any image rotation or scaling, and partially invariant to change in luminance or 3D camera viewpoint. There are four main procedures that compute and generate these type of features: Scale-space extrema detection, keypoint localisation, orientation assignment, and keypoint descriptor. Each section will be briefly discussed to understand its underlying process.

3.2.2.1 Scale-space extrema detection

Features of different sizes require different sizes of window to detect them. To do so, the Laplacian of Gaussian is used to detect blobs in various sizes with various scaling parameter. Lowe's(2004) implementation uses Difference of Gaussians instead to provide high effiency and speed. With the computed Difference of Gaussian, the local extrema over all scale and space is found and considered as potential keypoint.

3.2.2.2 Keypoint localization

The potential keypoints are then refined by a more accurate model to specify the extrema's location and scale. Keypoints are retained based on threshold on its intensity. Edge keypoints are discarded as well. This infers that only stable keypoints are retained for further computation.

3.2.2.3 Orientation assignment

Each keypoint location is then assigned with orientations based on the local image gradient directions. This achieves the invariance properties of a keypoint towards image rotation, scaling and orientation. Any further computation towards these keypoints will retain these invariant properties, increasing the stability of matching.

3.2.2.4 Keypoint Descriptor

A keypoint descriptor is then formed around the neighborhood of each keypoints. The keypoint descriptor is represented as a vector to allow significant levels of local shape distortion and change in illumination. The keypoints between two images are then matched by finding their nearest neighbours.

3.3 Door Detection

Some of the basic terms that will be used in the implementation of door detection is described as below. These are the definition of the described terms in Section 8.7.3

3.3.1 Linear Regression Fitting

Linear regression models the relationship between an independent variable and a scalar variable. The regression model used is the in the form of the line model, $y = ax + b$, where a denotes the gradient or slope of the model and **b** denotes the y intercept.

The gradient of the line model is calculated by the following formula:

$$a = \frac{(k\sum x\sum y - \sum x\sum y)}{k\sum x^2 - \sum x\sum x}$$

And the y-intercept of the line model is calculated by the following formula:

$$b = \frac{(\sum y - a\sum x)}{(k\sum xy - \sum x\sum y)}$$

Where the k denotes the number of points.

3.3.2 Colour Difference

The human eye is able perceive and differentiate between colours from objects. Although color spaces are able to describe a given colour in a mathematical way, detecting the change of colour requires the color space to be of linear characteristics to measure its distance. The area of colour science, particularly colour difference is the attempt to calculate the perceived distance between colours. Morkrzycki and Matol(2011) describe linear colour space as "the distance between the points defining individual colours being proportional to the perceptual difference between them".

One of the most common measurements used for color differences is the delta-E, which gather the differences for all dimensions in the colour space between two points into a single value. When calculating colour difference, delta-E uses the following formula:

Given two points $(R_1 + G_1 + B_1)$ and $(R_2 + G_2 + B_2)$, the difference in colour between two points, ΔE is calculate as follow:

$$\Delta E = \sqrt{(R_1 - R_2)^2 + (G_1 - G_2)^2 + (B_1 - B_2)^2}$$

4 DESIGN

This section discusses the reasoning and structured logical thoughts behind several important choices on the design of the system. Firstly, the method of data collection is discussed and justified, as well as any assumptions that had to be made. Then, both Image registration module and Door detection module will be discussed.

4.1 Data Collection

The chosen method of data collection is vital as the collected data must ensure that the context of the problem and the indoor environment are well addressed. Implementation approaches may also differ depending on the data collection approach as problems encountered are different.

Due to the limited distance of DepthSense311's depth sensor, a front view with a single vanishing point is not suitable as it cannot perceive any depth data further than 1 meter. The sensor will neither be able to detect doors along the hallway nor perceive enough to generate enough data for image registration. With this limitation, the camera must be positioned in a manner that obtain enough colour and depth data for the implementation.

As the project aims to detect doorways by identifying the difference in depth and the difference in colour as a doorway, the camera perspective is adjusted to focus only on the hallway walls. In this case, having the camera pointing down the hallway is not effective for the project's objectives as it does not capture the difference in depth in detail and is subject to influence of other objects in terms of identifying the effect of difference in colour on classifying doorways.

Taking a new approach to data collection method, the camera is pointed directly towards the wall in a perpendicular manner. Frames will then be taken along the hallway by moving in parallel to the hallway wall. When capturing frames, the distance between the camera and the hallway wall is constantly measured and kept at the same distance. The constant distance and fixed movement prevents any additional changes such as rotations or affine transformation to the frames taken.

Appendices 12.2 and 12.3 show example pictures of how the data are collected. In an attempt to be resourceful, a four-wheeled luggage is used as a trolley to slide across the hallway, with the camera on top of it pointing in a perpendicular manner. However, this method of collecting data means risking the problem of not being methodological and accuracy cannot be guaranteed. The only constant measurement being made is distance from the camera lens to the hallway wall is measured at 68cm before capturing the frame.

One of the obstacles faced during data collection is sensor aliasing. Sensor aliasing describes the non-uniqueness of sensor reading resulting in lack of information to produce any useful features, even in a noise-free environment.

Although not uncommon to the autonomous robot research area, Sensor aliasing presents itself with similar or identical geometry structure or lack of detectable features along a hallway.(Siegwart and Nourbakhsh 2004). In the context of our indoor environment and the choice of sensor used, the lack of significant features and designs made tracking frames captured along the hallway wall to be difficult. Sensor aliasing poses itself as a huge hurdle towards any further implementation such as image registration.

To counter this problem, Markers were placed along the hallway and frames are taken in a way that a marker will overlap between two frames. These markers act as distinguishable features for further processing such as stitching. Appendix 12.1 shows an example of the used marker, an augmented reality marker easily found on the internet. These markers do not contain any information in itself, but rather its obvious shapes and distinct colours prove useful as a feature to be placed along the hallway.

4.1.1Assumptions

To further reduce the dimensions of the problem, several assumptions are made towards all data collected. These assumptions deal with specific areas of image stitching and door detection that requires further implementation that may exceed the estimated implementation time.

4.1.1.1 Markers as artificial markers

Firstly, to maintain the integrity of the markers placed as artificial features, each frame taken contain two markers, with one on both side of the frame. These markers are non-unique, that is each frame contains identical markers on the left and right side of the frame. Hence, it is important that each frame must contain two markers to be used only as an identification of left and right side of the frame, and not imposing itself as a definite feature of the wall that can be exploited.

4.1.1.2 Sequence of frames are in order

Secondly, the sequence of the frames taken is presumed to be in order. Image stitching involving frames that are not in order adds unnecessary complexity to the project. There exists a whole sub-area of research in image stitching towards finding the right features and offset simultaneously that minimizes the error of image registration between images. At usual cases, these images contain unique features that can identify itself with another image. However in this case, each frame contains the exact same features, the markers placed along the wall. There isn't any perceivable difference between any two frames that has no unique identifiable features besides the two markers as the artificial features. Hence, it is important that the sequence of the frames taken is presumed throughout the implementation. This does not mean, however, that

the direction of the frames taken are not presumed nor known. The camera may be taking frames from left to right or from right to left.

4.1.1.3 Only one door in each case

Each frame collected may only contain at most one door. This is due to the simplicity of the implementation. Recognizing more than one door requires the door detection module to able to distinguish between doors and walls that are in between walls.

4.1.1.4 Three frames per dataset

With the rigid input/output library of C++, a certain number of frames that is loaded to the system is fixed. All dataset used with the system is presumed to have only four frames of colour and depth data.

With the fixed distance and perspective that the camera is capturing data of, neither the whole door frame nor both vertical edges of the door can be captured in a single frame. This leads to the need for image stitching, to create panoramic images that contain the door in a single image.

4.2 Image Registration

The goal of image registration is to find correspondence between two independent images of a scene. Image registration algorithms can be widely generalized to be feature-based matching or pixel based matching.

The correspondence found is a transformation that fits both images into a single coordinate system. Transformations are usually modelled based on the degrees of freedom, size of transformation, and the types of image characteristics it preserves. 2D transformations that are available are direct translation, Euclidean translation which includes rotation, affine transformation, projective transformation and scaled rotations.

4.2.1 Parametric Motion Models

Recall that before any image registration can be done, it is important to determine the motion model of the two images. Motion model is described in Section 3.1.1. To minimise the level of complication in image stitching, all image registration done involves only finding **2D planar transformation** between two 2-dimensional images. This means that no 3-dimensional or spherical transformations are involved, and any image alignment performed was purely translational. To be able to do so, the methodology of data collection must have this issue in regards. Capturing data in a perpendicular view and a parallel movement as described in Section 4.1 ensures that there is less degree of freedom being dealt with during the implementation.

It is wise, however, to understand that there can be no clear guarantee of zero rotational, perspective or affine transformation involved during the event of data collection. That is, any transformations besides planar transformation are not explicitly dealt with in the implementation, with the assumption that it does not have high influence or addressed in the implementation of image stitching algorithm. This clearly does not mean that any other motion models can be ignored to achieve the optimal performance, but dealing only with simpler transformation means reducing the scale of problems needed to be addressed.

4.2.1.1 Scaling Depth Data

The first image registration done is to match the resolution of depth data to the colour data. This is done by scaling depth data to match the resolution and size of the colour data. The transformation found is a direct scaling. Although to synchronize the depth data with the colour data, the instructed steps is to first extract the U and V coordinates of a depth point from DepthSense API's UVMap, then with these coordinates to locate the right colour pixel value. However in this implementation the need of scaling the depth data is only to match hard edges and not to create a coloured point cloud. The resampling of depth data is done by OpenCV *resize()* function which uses nearest neighbour algorithm for interpolation.

4.2.2 Image Preprocessing

A pre-processing phase is included in the process pipeline to improve the results of image registration. DepthSense311 has a relatively low resolution; hence noises are apparent and may affect the performance of algorithms that work directly on them.

4.2.2.1 Color Image noise

The colour data captured contains hue noises which may affect detection performances. To eliminate these noises, Gaussian blur was applied to the color data before being passed down the pipeline.

Due to the lack of features and no feature-based image alignment algorithms are used, de-noising kernels such as Gaussian kernel will not have a big effect on the integrity of the data. The only modified variable in the applied Gaussian blur is the kernel size, which was set at 13x13, while the standard deviation of the 2D Gaussian was calculated based on the kernel size. The calculation of the standard deviations is done by the OpenCV library.

4.2.2.2 Depth Image noise

Also due to DepthSense311's low resolution, depth data captured contains artefacts that will have negative effects on further image alignment implementations. These artefacts are depth values that cannot be captured by

the camera, and were marked as undefined. To reduce the effects of these artefacts, a median blur is applied to all depth data frames. Median blur replaces a pixel value with the mean of its neighboring pixel values within a kernel. The kernel size applied is 3, and its implementation is done using the OpenCV library.

4.3 Pixel-based Image Alignment

When trying to find the correspondence between two images, there are generally two categories of image alignment techniques: feature-based alignment, where the positions of significant features found in the first image will be matched in the second image; or pixel-based alignment, where differences between direct pixel values are calculated between two images, and a transformation that minimizes the differences will be the chosen.

Pixel-based alignment has advantages over feature-based alignment techniques in that it provides more precise and accurate alignment due to its direct approach to access and evaluate pixel values. In this case, there are minimum features in the hallway that can be used for feature-based alignment. In fact, to solve the sensor aliasing problem, markers must be placed along the hallway as significant features for any image alignment techniques to work.

4.3.1 Error metrics

The normalized cross-correlation is used to find the offset for the translational alignment. As described in Section 7.1.3, this error metrics calculates the correlation values between the two images. With normalization, the translational alignment is computed over a normalized exposure, where two images' exposure is averaged out to avoid exposure differences affecting the correlation value. The offset coordinate with the highest normalized cross-correlation value indicates that the two images have the highest correlation if it overlaps at that coordinate. The coordinate will then be used for stitching between the two images.

During the process, the seam between the two images are ignored and not dealt with. This is because there isn't a need for a clear seamless panorama image. The implementation also does not deal with sub-pixel accuracy, which will complicate matters further by needing interpolation of data if the offset is fractional.

4.4 Windowed Image Registration

With relatively large images, the process of calculating the normalised Cross-Correlation can take a very long time for going through all pixels between two images for all possible displacement values. Although accurate, it is neither practical nor sensible to calculate the correlation for all pixels between two images. To reduce the size of the search space, only certain patch of the image is involved in the calculation.

As addressed on Section 7.1, one of the weaknesses of pixel-based image registration is that the difference between pixels minimized may not be the optimal offset for stitching. This is because features in an image are often unevenly distributed. In addition to the fact that only an area of the image is involved in the algorithm, the correlation may not accurately represent the

correct solution. Hence, it is extremely important that the chosen windows to perform pixel-based image alignment are at the ideal area, with the ideal size. To determine that, certain Information obtained from both depth and colour data can be used to provide further insight into getting the correct stitching offset. Any features available should be used to provide the accurate window for pixel-based stitching. The artificial markers, which are presumed to exist in both side of a frame for all frames, can be a good lead to where the window should be.

4.5 Marker Detection

To locate the markers in the frame, an object detection technique is used to find the region of interest. SIFT features are extremely useful in this case to detect the location of the markers. SIFT features are invariant to scaling, rotation, translational transformation in any image, as described in Section 7.2.2. OpenCV provides a simple implementation guide to locating SIFT keypoints of a given object in an image using its library. A descriptor first computes the SIFT features available in the given template image as well as the scene image. With these features, the maximum and minimum distances between these keypoints are then computed, and only features that passed a certain threshold are considered as "good" keypoints that matches with the template images' SIFT keypoints.

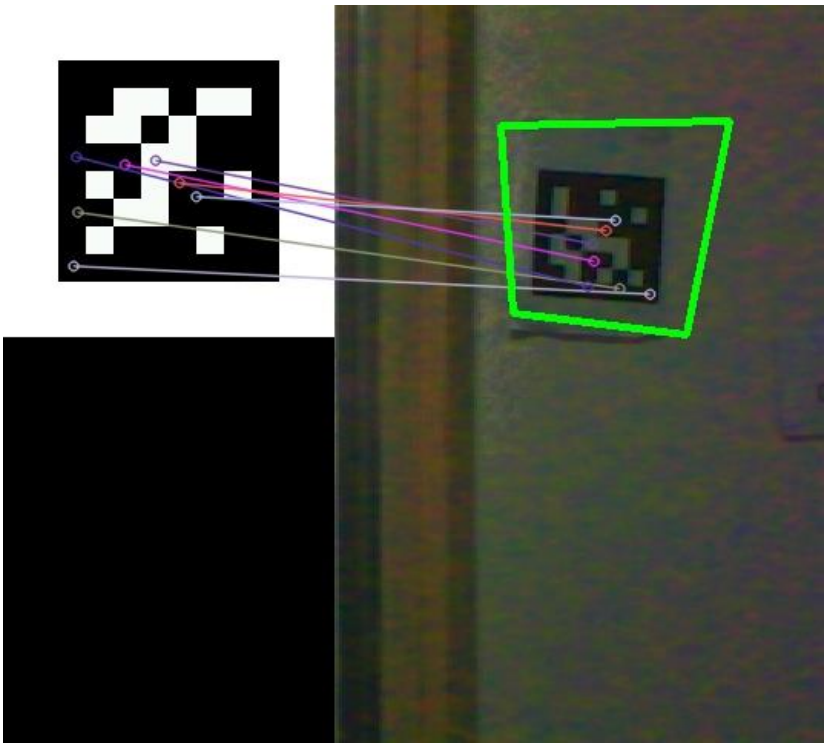


Figure 8.1: SIFT feature detection that matches the marker template image to the scene image

With these filtered keypoints, a homography can computed to find the transformation model from the template image to the scene image. Finally, the homography provides a model to perform perspective transformation on the

template image onto the scene image. The transformation locates the four corners of the keypoints, and an area of interest can be extracted out of it.

Figure 8.2: The extracted marker using detected SIFT features

However, the above implementation does not go well for locating multiple objects in the image. Having multiple objects implicate that more "false" keypoints matches will be computed. The keypoints matched may be mixed between the two markers, where some keypoints are matched to the first marker and others to the second marker, as the feature detector is not able to determine if which marker each keypoints belongs to. With the given presumption, the solution to this hurdle is to split the image into the left side and right side of the image, where each markers can be individually be extracted and the locations of these markers can be stored. On the flip side, this means that for each frame SIFT features are computed twice, making the implementation needing more CPU and longer processing time.

Knowing where the markers are, the image alignment problem can now reduce to whether normalized cross-correlation should be applied to the left or right marker of each frame. Since there is no presumption of the direction of the frames taken, the first image to be registered must determine the direction for the rest of the registration pipeline. One of the methods to decide is observing any differences in colour and depth in both images.

4.6 Depth Deviation Detection

One of the more prominent information that can be used is the change in depth. A change in depth usually indicates an indentation along the hallway. This change in depth can be used to determine which marker is overlapping with the next frame. When the measured depth of one marker is similar to the next frame's depth value while the other is not, we can infer that the overlapping area is in that marker's region.

The change in depth is detected by having a sliding window set at the left end of the frame. The standard deviation of depth in that window is calculated and recorded at the pixel in the middle of the window. The sliding window then slides a pixel forward and the same process is repeated. This generates a standard deviation graph, with the peak in the graph indicating a change in that

area. All peaks above a fixed threshold are then extracted and stored in a custom class that relates to that frame.

One of the neat features of the standard deviation of the depth graph is that the extracted depth excerpt captures the edge, or the gradient of the graph. Splitting the captured window will divide the window into two different regions of different depths, where one region on the left has a different depth compared to another region on the right. With the help of these two different regions, it opens up the possibility for more accurate choice which marker should be chosen for stitching. These regions are then inspected based on its colour data.

4.6.1 Colour Segmentation and classification

The change of depth is an interest point that can be further strengthened if there is also a change in colour. One of the symmetry in the data collected is that in each overlapping area between two images, the colour should always agree or have minimum dissimilarities. With that information, each frame can be segmented into its colour, and based on the symmetry in colour, it can be determined which side of the frame is to be stitched with. Colour segmentation is performed on an area of interest determined by the deviation in depth value. A change in depth is prioritised over a change in hue as obstacles or other objects with different colours on hallways may be misleading, assumed as a possible keypoint for a door frame.

The windowed area can be reduced into its two dominant colour using K-means clustering. K-means clustering finds two pixel values that can be used as the general classes that can separate all pixels into two groups. Once the two dominant colours are found, it is stored and can be compared with the second image's windows' dominant colours.

With the consistency in depth data and colour data, we can determine the direction of stitching, whether the first image stitches with the second images in a leftward or rightward direction. Once decided, the rest of the images use the same pattern for stitching. Recall that one of the assumptions made is that all frames collected are in an orderly sequence, allowing quick stitching to be done.

With the given information, the collected data can now be stitched into a single panorama image as shown below. The panorama depth and colour image will then be used for door detection.

4.7 Door Detection

The implementation of door detection strikes similarity with the image registration component. This is due to the limited features and information that can be used. The algorithm design draws inspiration from Hensler et al.(2010)'s implementation, which

combines several weak classifier with AdaBoost Learning Algorithm to produce a strong door detector classifier. A region is classified as a true positive in the door detector if it possesses indentation along the wall and a change in colour where indentation happened.

4.7.1 Depth Data Analysis

The first phase of detecting a doorway is to identify the hallway walls and door. Depth data taken from the sensor are used to identify the existence of a hallway wall. Opting for Occam's Razor, the hallway wall is modeled by a linear regression line in a 1-dimensional depth image rather than a plane model in a point cloud. This reduces the needless extra dimension, where the height of the wall does not provide useful information for detecting doorways at all.

4.7.2 1-Dimensional Image Analysis

The depth data is represented by a 1-dimensional image that is able to show the average depth data of the image in every point. The average depth data is calculated by first calculating the mean of all the depth values for each column in the depth frame and assign it as the value for that row. This way we can identify an even plane in the depth data by fitting it with a line, and any rise or drops in the values means a second line can be fitted, indicating a dent on the wall, presuming to be a doorway. Figure 3.1 shows an example of 1-dimensional depth image visualized to show the changes in the depth in the image, and its corresponding 2-dimensional depth image in Figure 3.2.

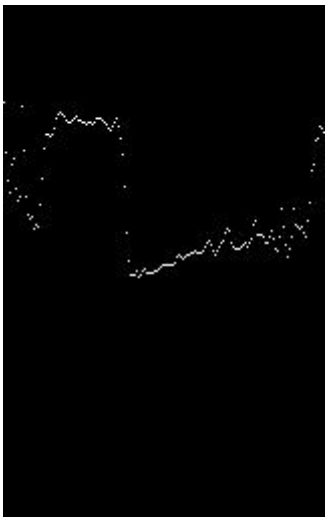


Figure 8.3: An 1-dimensional depth graph image visualized to show a dent along the hallway

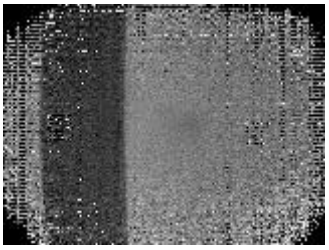


Figure 8.4: The corresponding raw 2-dimensional image.
The corners and white noises are sensor artefacts

The change in depth is identified by the change in standard deviation along the depth graph. Similar to the depth deviation part in image registration, a sliding window slides across the depth image and calculates its standard deviation each step. The goal is to find the window with a sudden change in standard deviation, and split the image into two regions, one before and another after the window with a steep gradient of standard deviation.

These regions are then fitted with a linear regression line to find the available line models in the image. The line models found within these regions are then stored for comparisons. Considering that these line models are only used to differentiate planes of different depth, the gradient of the line models do not hold as large importance as the y-intercept does. A line model with a smaller y-intercept represents a section of the wall that is deeper than a line model with a higher y-intercept.

4.7.3 Line Model Excerpts

These line models, however, can be redundant as they are computed using the regions extracted based on any depth deviations. Several line models may be closely related or have minimal y-intercept differences. Anguelov et. al(2004) mentioned how failing to recognizing wall segments as a single object may lead to poorer mapping, hence the next step is to reduce the number of line models by converging these related line models into a common line models with significant difference in y-intercept. Line models with similar y-intercept values are combined, and the line model with gradient closer to zero is chosen.

Figure below shows an example of the final extracted line models computed for the given depth panorama image. The line models successfully represent the available planes in the data based on its depth. Although it does not reflect a perfectly flat plane for the hallway walls with accuracy, the line model satisfy the requirement of segmenting the wall based on indentation. The line models for the wall are not joined together for the clearer representation of indentation. These line models can now be used to search for possible door candidate among the excerpts.

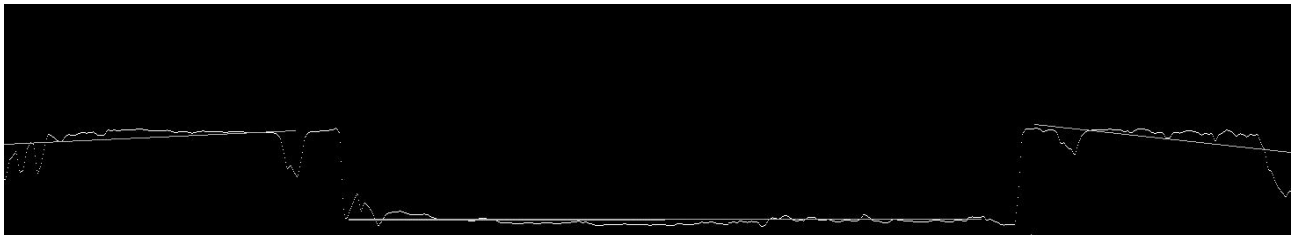


Figure 8.5 shows a stitched depth graph with three line models found.

Note that in the image the line models drawn did not include the areas of the graphs with sudden drop or rise in depth. These changes in depth do not need to be shown the line models as they do not contribute to the plane representation. Future works that can be done is the implementation of piecewise linear regression, connecting the two line models with a third line model that represent the gradient.

4.7.4 Colour Segmentation

With the extracted excerpts, the dominant colours of each excerpt are then compared. As each excerpt are in sequence, it is possible to detect a change in colour for the excerpts. A change of colour is calculated by the colour difference delta-E, as described in Section 7.3.2, in the RGB-colour space. This means that the colour difference calculated does not actually taken into account the actual perceptual difference. If the colour passes a fixed threshold, the colour difference is considered as perceivable difference and will be taken account into door classification. When a region is detected to have a large colour difference with its neighbouring regions, the region is considered to be a door candidate.

4.7.5 Text recognition

With the extracted door candidate, any textual information is essential to recognizing it as a doorway. A pre-built text recognition library is used to extract any texts available on the given door candidate excerpt.

5 IMPLEMENTATION

This section discusses the reasoning and structured logical thoughts behind several important choices such as platform, programming language, hardware, third-party libraries as well as core implementation. Problems and conflicts encountered along the way are also discussed at the end of this section.

5.1 Platform Choices

The chosen development environment must be flexible, quick and widely supported. This ensures that the project will not be delayed by unfamiliarity of programming language, unknown issues from third party libraries, and implementations will not be

limited by a small API. The Operating System used for the development is Windows, Intel i3 Processor with 6GB of RAM.

5.1.1 Programming language

The chosen programming language is C++. It has a fairly common and standard language notation, widely supported for many computer vision libraries, and supported by the choice of hardware. The **downside** of it is easily being complicated as the software grows in size with more namespaces and classes. Memory and garbage management are manually done and must be paid great attention to. Writing GUIs can also be somewhat tedious and complex.

5.1.2 Integrated Development Environment

The chosen Integrated Development Environment (IDE) for the project is Visual Studio 2012. Visual Studio is chosen ahead of other IDEs such as Eclipse and NetBeans for familiarity. Visual Studio is common for C++ support and provides extensive debugging features. The downside of Visual Studio is integrating third party libraries where linker errors and complex differences in debugging and release mode making it difficult to setup and configure.

5.1.3 Hardware choice

The sensor that was chosen for development is DepthSense 311. DepthSense 311 is a depth-sensing camera (rgb-d) similar to Windows Kinect. DepthSense camera was chosen ahead of the Kinect camera mainly due to the cost factor, pricing at only at a third of the Kinect's price point. DepthSense311 also allows depth data gathering powered by USB at a shorter distance, but plugging it into a power source allows it to gather at a farther distance.

However, DepthSense311 has lower resolution compared to the more expensive Windows Kinect. It supports only 120x120(QQVGA) for depth data and 640x480(VGA) for its color data. Albeit its low depth data resolution, it does not affect implementation effort as high details in depth data aren't necessary for this project.

5.2 External Resources

Third party libraries are used to increase development speed as many of the common algorithms are supported and well written. This helps to negate the burden of implementing them from scratch, risking making low level mistake leading to hidden bugs and errors in the future. Although third party libraries allow simpler and higher level implementation, the underlying algorithms are still read and understood before using them. Only open source third party libraries are used, ensuring there's no legal conflict while maintaining a strong support from the community.

5.2.1 DepthSense SDK

The DepthSense SDK is the official API that allows interaction with the DepthSense311. The SDK is neither complicated nor large in size. The main purpose of this library is to allow simple retrieval of raw data such as color data, floating vertices point, registration points map, and timestamp. Although the DepthSense SDK is not open source, it is required to interact with the hardware sensor.

5.2.2 OpenCV

Open Source Computer Vision (OpenCV) was developed by Willow Garage as an open source library of C/C++ functions that covers many image processing and real time computer vision algorithms. OpenCV is very popular for its speed, portability and requires less resource. It is also widely supported by an active and strong community that continues to improve the library.

5.2.3 PCL

The Point Cloud Library (PCL) is a cross-platform, open source, large scale project that allows point cloud processing. PCL contains state-of-the-art algorithms that deal with point clouds such as filtering, feature estimation, registration, model fitting and segmentation. PCL is also strongly supported by users from dedicated mailing list.

5.2.4 Tesseract

Tesseract claims to be one of the most accurate open source OCR engine available. Combining with the Leptonic Image Processing Library, Tesseract is able to read over many different types of images and converts its text to more than 60 languages. Despite its strength, Tesseract received little improvement over the years and had been releases had been slow with a focused group of community involved in it.

5.3 Graphic User Interface

Being a research project at its core, the developed graphic user interface uses a command line interface. The interface starts by prompting the user for entering the folder that contains the required images. Once entered, the colour and depth images will be read from the entered folder into the system. All colour and depth frames must be labeled in the following format:

Colour frames ; <index>cframe.jpg

Depth frames : <index>depthframe.jpg

With the index beginning at 1, only three pairs of images will be read into the system. The system then begin image registration module. Panoramic images provided by the image registration module will then be feed into the door detection module.

With the given panoramic images, the door detection module will identify and notify the line models found in the depth panoramic graph, with it the gradient and y-intercept for each model. Finally, if a door candidate is found, the colour excerpt will be shown together with its depth graph excerpt and its colour.

5.4 Data Collection

The colour and depth data are captured using DepthSense API and a demo file provided that communicates with the DepthSense 311 camera. As the colour image is only available in HSV color space, the data is first converted to RGB values before storing into OpenCV's Mat format. On the other hand, the depth data stored is directly from the camera. The values stored represent the vertices points of the depth sensor, which calculates the distance to the target object in millimeters.

5.5 Image Registration

Image registration is performed once all colour and depth frames are loaded into the system. It is performed onto two frames at a time, namely a previous frame and a current frame. A Frame class stores all useful information such as the dominant colour of the frame, the location of the markers, regions of deviation in depth for both depth frame and colour frames, and the raw colour and depth data.

Firstly, the location of the markers are found using OpenCV's Surf Descriptor that implements a SIFT algorithm. The library implementation requires an upload of a template of the marker, which was specified and provided to the system, to calculate invariant keypoints in both the template and the image. These keypoints are then matched based on a nearest neighborhood calculation to locate the marker. For both frames, the left and right markers are located and stored in the Frame class.

Then, the direction of the stitching needs to be determined. The direction need only be determined at the beginning of the module. Once identified, the rest of the image registration can be done according to the sequence of the markers. A function call *findWindowInterest()* Looks for the direction of the stitch based on the consistency of depth values and colour. Recall that a Frame class contains the depth deviations of the given frame. With this, we can compare and determine the consistency of the depth values. The last deviation of the previous frame and the first deviation of the current frame is first compared. As deviation windows saved in Frame class is a change of depth in the depth data, splitting the window down the middle splits it into two regions of different depth value. Then using *getWindowDepthValue()* the average depth value of the split region is calculated and compared.

Note that it is false to have both frames to have deviation windows at all cases. At times when only previous frame had a deviation window, the last deviation window

is checked with the exact same location of the other frame. At this circumstance the location of the window in the other frame does not matter as there is no deviation in depth throughout the whole frame. Once a consistent depth value is found (difference in depth is less than 40%), the direction of the stitching can be determined.

If both frames had deviation in depth, the same iteration is done as pervious cases, but the difference in colour is checked as well. If the difference in colour is less than a certain threshold, the deviation in depth will be checked. This ensures that the stitching will be based on a consistency of depth and colour.

The actual stitching is done by the *stitch()*function that calculates the required space in the OpenCV's Mat format and the offset value. Offset value may differ if one image is bigger than the other. Commonly this is dealt with bundle adjustment algorithms, but simple calculations suffice for the requirements of this project as only 2D translational transformation was involved. The finished panoramic images for both depth and colour image are then passed to the *DoorDetector* class module.

5.6 Door Detection

Upon receiving the panoramic images, the first process is to calculate all possible line models in the depth image. *displayDepthGraph()*converts the panoramic depth image into a depth graph, where deviation windows in depth will be calculated again with *searchDepthDeviation().*The provided deviation windows are then used to find all possible line models with *findDepthLines().*Taking the same trick out of the book, the deviation window are split in the middle into two regions of different depths. These regions are joined together with other regions of different deviation window to form a segment.

A segment is an extract of the *depthgraph* that are between two deviation windows. The reasoning behind this is that it can be assumed each segment has the consistent depth, and a panoramic depth image may have 1 or more segments. These segments are then fitted with a linear regression line and the line model is stored as well as the segments are stored.

The stored segments can be used to detect the change of colour. As they had different line models, each segment are of different depth beyond an acceptable threshold. If there's a segment that was discovered to have a difference of colour between two segments, the segment is extracted as the door candidate. All information such as the line models, the segment, and the depth graph are then stored in a *DoorCandidate* class and returned to the system. The colour segment is then used to detect any available texts using the Tesseract library, which provides a text recognition function, and stored into the *DoorCandidate* class and printed to the user interface.

6 RESULTS AND EVALUATION

6.1 Data Collection

The evaluation of the system is divided into two parts: Image Registration and Door detection. Each component is provided with different types of data collected to show the strength and weaknesses of each component. All dataset collected have images of colour and depth frames. They differ from each other in terms of indentation, colours, and types. Some datasets are doors that have indentation but not change in colour, while others are wall indentations that are not actual doors.

As the system is engineered towards always producing a door candidate, negative results are not explicitly produced by the system. Instead, the system should be evaluated based on its positive results as a display of its accuracy and reliability. The justification for this approach is the fact that a door module, when integrated into an autonomous system, do not need to constantly scan and calculate false negative results, but to pick up the existence of doors along a hallway.

The evaluation of the system is not based on a quantifiable or statistical approach. The underlying implementation is:

1) Not robust enough to detect a large number of doors effectively as it only consider a difference in depth and difference in colour. Despite it being a door detection system, the results of a statistical analysis will not prove to be useful nor reflective as it does not justify nor criticize the use of difference in depth and difference in colour as the consistent feature that can be related to detecting a doorway.
2) The aim of the project is to understand the consistency of certain features relating to doorways, which is implemented in a broad sense with the difference in depth and the difference in colour.

6.2 Image Registration

With minimum transformations in the motion model presumed, the image registration module is evaluated on the viability of the method in which the data is collected. For recall, the data collection is done with a low resolution RGB-D camera which is pointed perpendicular to a wall and capturing frames in parallel to the wall. This minimises the motion model to the simple 2D translational transformation. The evaluation requires the markers in the frames to be stitched accurately regardless of any inconsistency between exposures and motion models of the frames.

Figure 10.1 : Exemplary result of image registration despite exposure inconsistency

The figure above shows an exemplary result that performed image registration for four frames accurately despite the inconsistency of exposure. Although there are no definite wall indentation captured, the image registration module is able to successfully determine the direction of the stitching and locate the markers correctly. Original individual frames of this figure can be found in the Appendices.

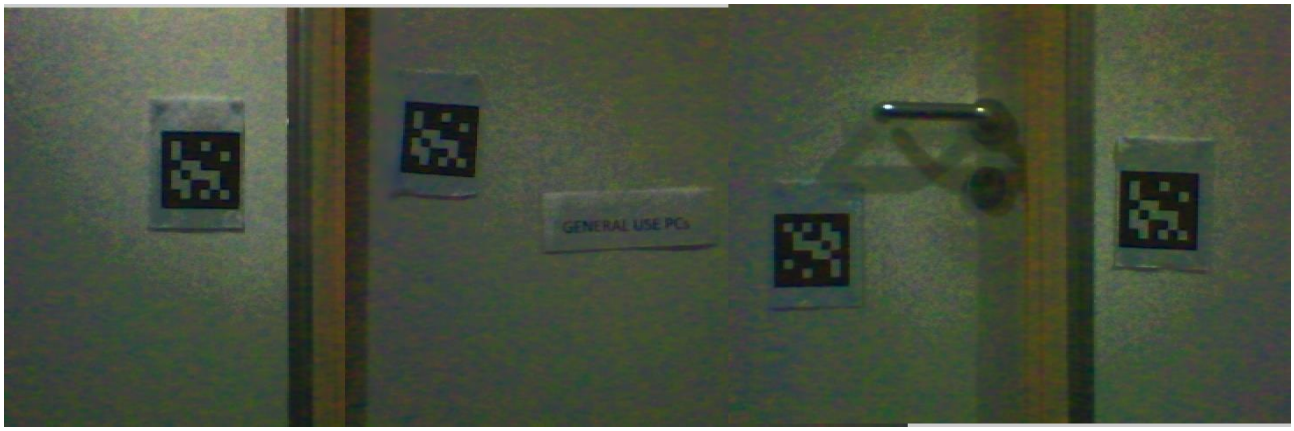


Figure 10.2: Example of Successful stitching

The figure above shows another result where only a shallow difference in depth and minimum changes in colour does not affect the stitching of the image. Similar to Figure 10.1, the image registration module is again successfully determined the direction of the stitching and locate the markers correctly. Original individual frames of this figure can be found in the Appendices.

6.3 Door Detection

The implementation of the door detection module heavily focuses on two main properties: Indentation along the hallway and change of colour. These two properties are tested in various environment to test if they play a major role in detecting the presence of doorways accurately.

6.3.1 Line Model Accuracy

A successful line model function should be able to extract the minimum but correct amount of line models from the panoramic images. However, that may not always be the case as shown with the results below.

Figure 10.3, 10.4: Example of redundant line models on depth graph of example dataset with its panoramic colour image as counterpart.

The graph above is the representation of the panoramic depth image of the colour panorama above it. The system found at least 5 but filtered down to 3 distinct line models, and despite having two indentations, the line models failed to recognize it as separate segments and only found one single deviation window. This may be due to the threshold of the window being too high, which in turn only one segment of the provided line model is extracted. Due to this error in addition to the difference in colour between the two segments, the system was led to a false classification of a door candidate as shown in the figure below. The screenshot of the system can be found in the Appendices.

6.3.2 Colour Difference

Figure 10.5, 10.6, 10.7: Example of successful door candidate extracted with its panoramic colour and depth image as counterpart.

The figures above show an example dataset that was successful in extracting a doorway. The combination of difference in depth and difference in colour makes a section of the panorama a clear candidate of doorway. Note that from Appendix 12.4, there are three filtered line models found, with two line models having similar y-intercept values, at 146 and 141. The extracted segment had a largely different value at 230. The difference in y-intercept indicates the change in depth.

Figure 10.7, 10.8: Example of failure to extract candidate due to similar colour between door and wall

The figure above shows a situation where the change in depth in definite and clear, but due to the consistency of colour, no door candidate was found by the system. Notice how similar the depth graph of this dataset is to the previously mentioned dataset. This is a clear exemplar of the system being unreliable in decision making based solely on difference in depth and difference in colour. However, further improvement can be done. There may be cases such as this where the door is of consistent colour with the hallway walls, but the frame was designed to be of different colour, and further implementation on the difference of colour in frames as one of the perceivable properties of a doorway can be explored.

6.3.3 Text Recognition

The text recognition provided by Tesseract was not successful in detecting any text when tested on actual door candidates. The implemented function failed to provide any sensible results nor texts. Appendices include the images and the result text recognition of door candidates.

7 CONCLUSION AND REVIEW

The door detection in hallways is a rather large yet mildly explored area. It holds an important position in the future of SLAM systems in indoor environments and should be encouraged to further research into a more robust door detection module.

In summary, although the project do not hold against the standards of a quality software, with reliability, accuracy, and security, it was able to show determine that the

difference in depth and difference in colour as some of the constant properties of a doorway. The system was also able to successfully extract candidates that had a difference in depth and a difference in colour. However, the failure to recognize texts could be improved. One of the reason text recognition did not worked may be due to the low resolution of the image.

The results of the implementation had been far from satisfactory, as reliability of the implementation can be heavily improved. Many functions that do comparisons such as colour comparison or depth deviation uses a fixed threshold, which is not sensible as the value does not hold true for all different environment and exposure. The extraction of line models can also be improved with piecewise linear regression, where two line models can be computed instead of one single regression line.

One of the insights gained from the project is that with additional information such as robot's odometry, the perspective used for data collection can be a useful, efficient and low cost method for data collection. For recall, the perspective used was facing the camera in perpendicular towards the wall while capturing frames in parallel to the wall. Image registration that only consider 2-dimensional translation motion model can greatly decrease required time. Low cost rgb-d camera reduces the cost of implemented systems, and with the identified doorway properties, door detection module can be fast and possibly online.

7.1 Future works

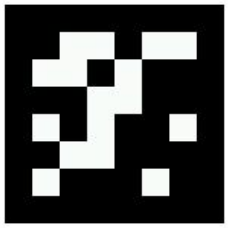
Besides improving the current implementation that focuses on difference in depth and difference in colour, there are a more properties that can be investigated regarding its consistency as a doorway's property.

In terms of object recognition, the detection of door knobs of different kind can be explored. Being able to detect door knobs can be very helpful in quickly recognizing the existence of a doorway. However, open doors should also be taken into consideration regarding to door knobs.

Any detected texts can be interpreted with Natural Language Processing to infer any information regarding existence of an enclosed space, namely a room. Room labels are a common source of information to study an environment. Being able to recognize and understand the context of the text increases the likelihood of detecting a doorway and exterminating any false doorway candidates where text is present.

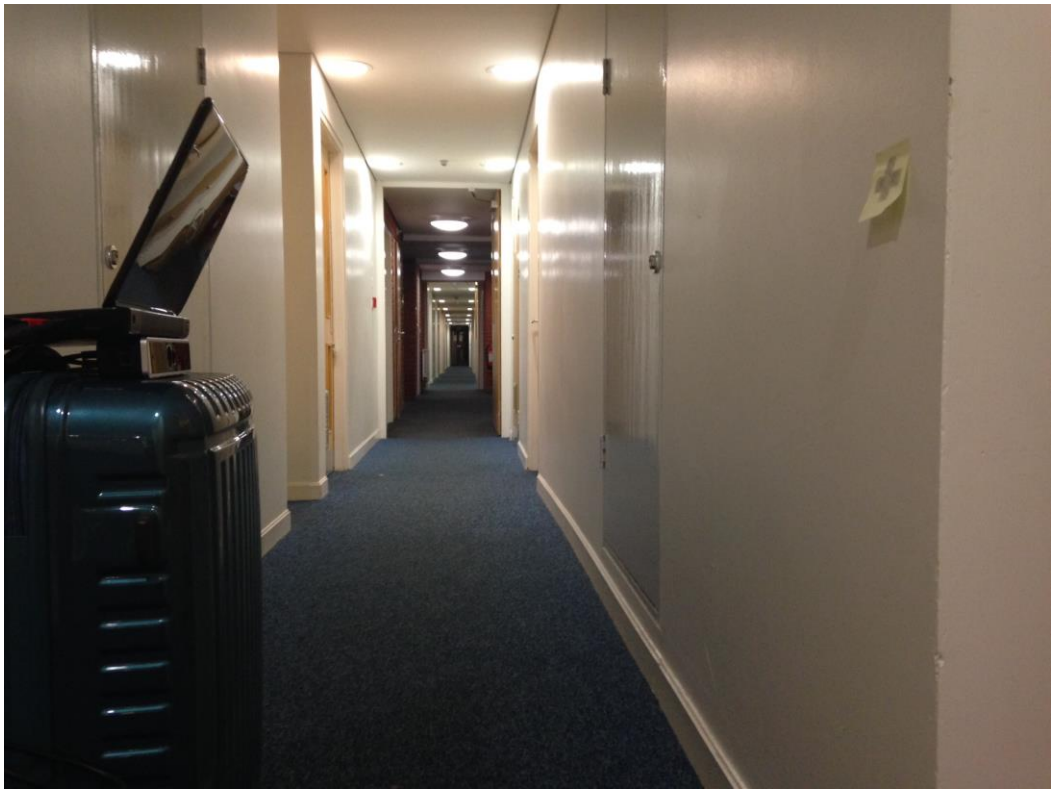
For more complex implementation, the probabilistic generative model approach can generalise the characteristics of a door in an indoor environment. As doors in a specific indoor environment are typically possesses consistent characteristics, the ability to detect and learn the door model of a specific environment is more reliable for different indoor environments.

8 APPENDICES

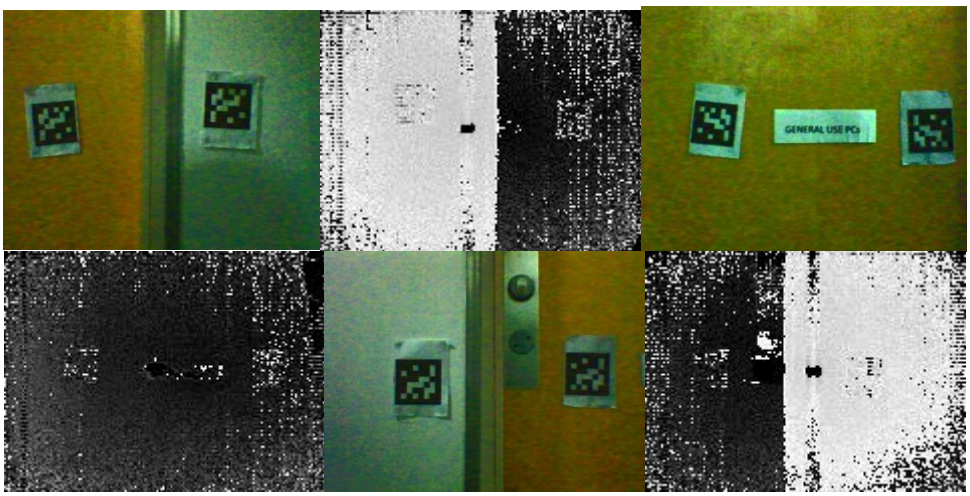


Appendix 12.1: The original image of the marker used in data collection

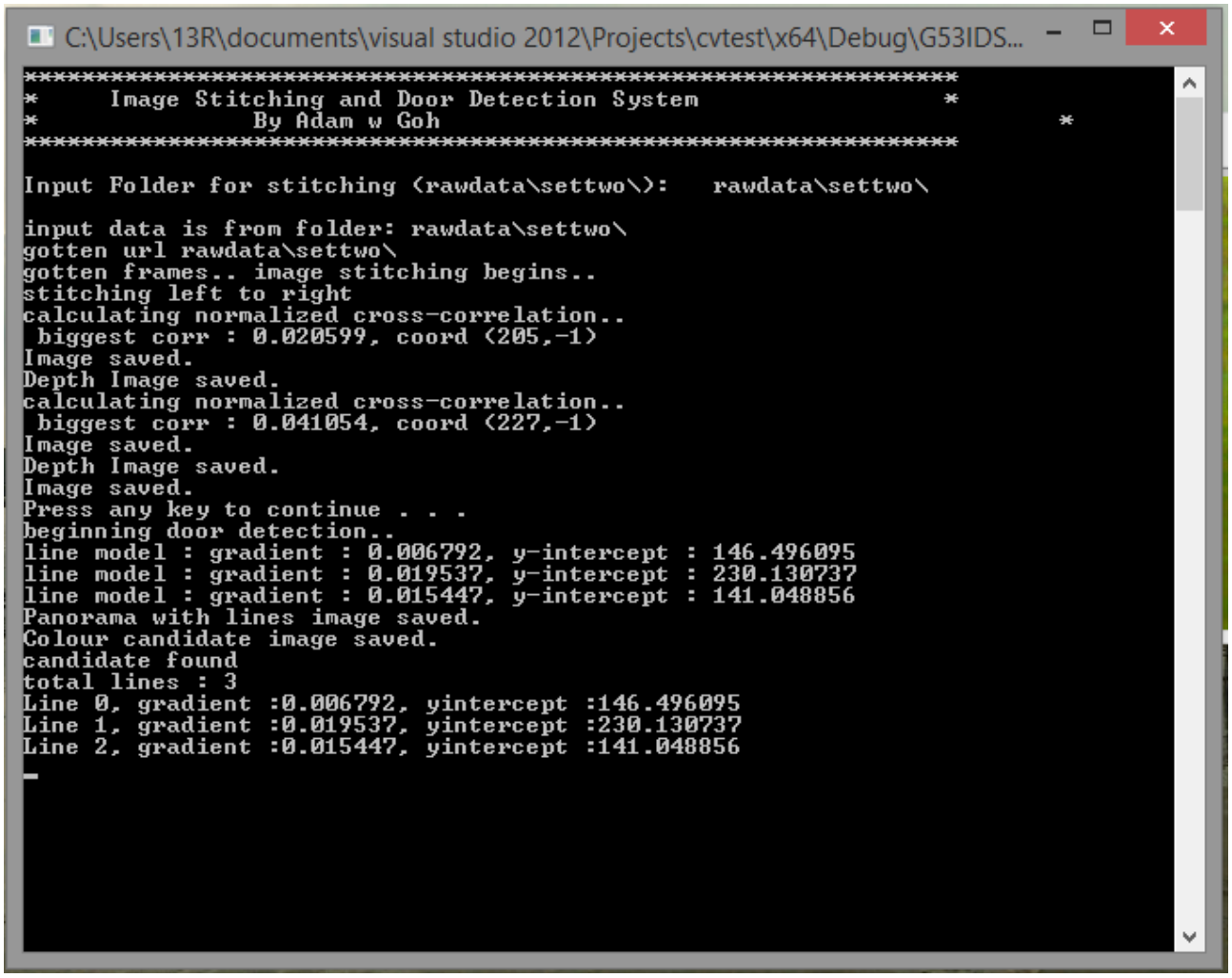
Appendix 12.2: Example situation of the data collection process



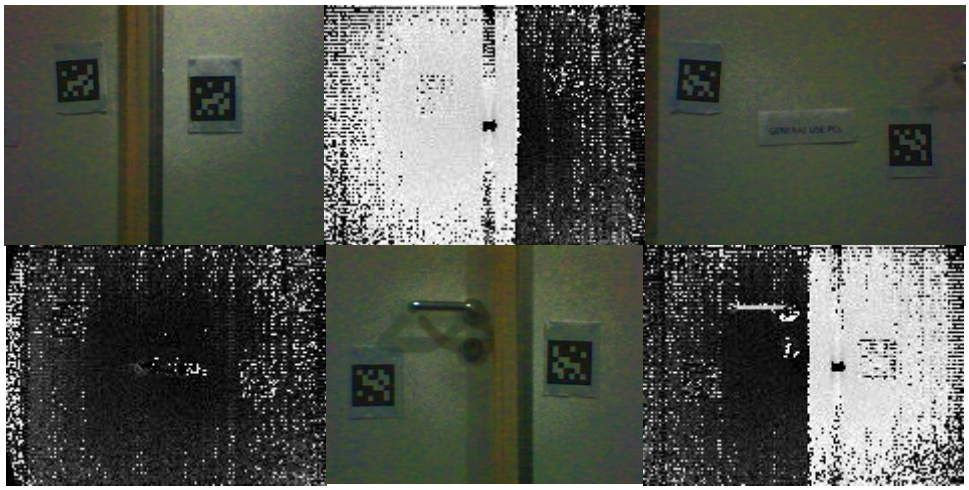
Appendix 12.3: The arrangement of camera setup. The distance between the camera to the wall is measured at 68cm.



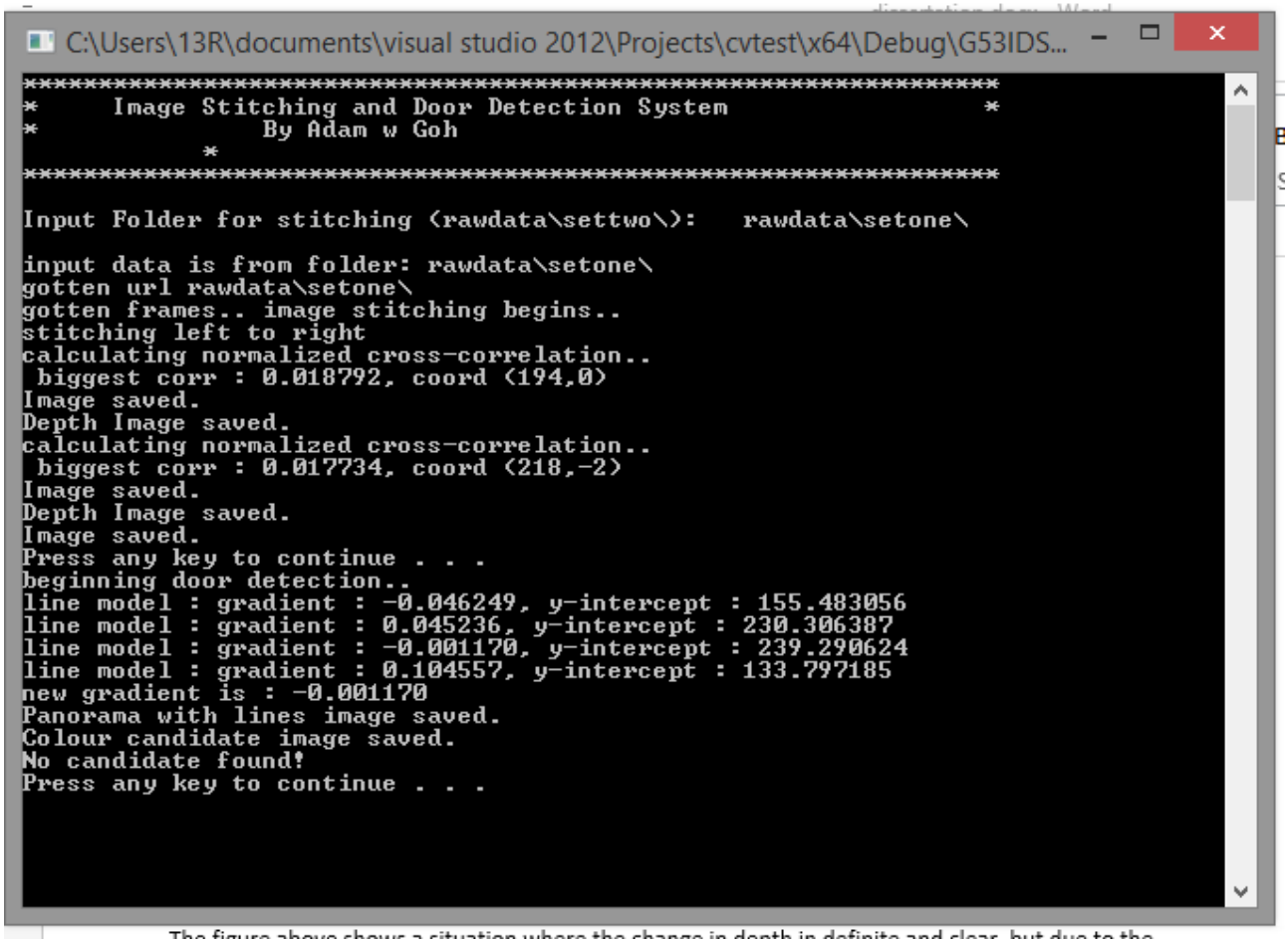
Appendix 12.4: Raw colour and depth data frames collected from the camera.



Appendix 12.5: The results of the system for the dataset in Appendix 12.4



Appendix 12.6: Raw colour and depth data frames collected from the camera.



Appendix 12.7: The results of the system for the dataset in Appendix 12.6

Appendix 12.8: Raw colour and depth data frames collected from the camera.

9 BIBLIOGRAPHY

Anguelov, D., D. Koller, E. Parker and S. Thrun (2004). Detecting and modeling doors with mobile robots. Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on, IEEE.

Derry, M. and B. Argall (2013). Automated doorway detection for assistive shared-control wheelchairs. Robotics and Automation (ICRA), 2013 IEEE International Conference on, IEEE.

Ekvall, S., D. Kragic and P. Jensfelt (2007). "Object detection and mapping for service robot tasks." Robotica **25**(02): 175-187.

Fernández-Caramés, C., V. Moreno, B. Curto, J. Rodríguez-Aragón and F. Serrano (2014). "A Real-time Door Detection System for Domestic Robotic Navigation." Journal of Intelligent & Robotic Systems **76**(1): 119-136.

Garulli, A., A. Giannitrapani, A. Rossi and A. Vicino (2005). Mobile robot SLAM for line-based environment representation. Decision and Control, 2005 and 2005 European Control Conference. CDC-ECC'05. 44th IEEE Conference on, IEEE.

Gil, A., Ó. Reinoso, M. Ballesta and M. Juliá (2010). "Multi-robot visual SLAM using a Rao-Blackwellized particle filter." Robotics and Autonomous Systems **58**(1): 68-80.

Gilliéron, P.-Y. and B. Merminod (2003). Personal navigation system for indoor applications. 11th IAIN world congress.

Hensler, J., M. Blaich and O. Bittel (2010). Real-time door detection based on adaboost learning algorithm. Research and Education in Robotics-EUROBOT 2009, Springer**:** 61-73.

Kim, D. and R. Nevatia (1998). "Recognition and localization of generic objects for indoor navigation using functionality." Image and Vision Computing **16**(11): 729-743.

Lee, J.-S., N. L. Doh, W. K. Chung, B.-J. You and Y. I. Youm Door detection algorithm of mobile robot in hallway using PC-camera.

Lee, J.-S., N. L. Doh, W. K. Chung, B.-J. You and Y. I. Youm Door detection algorithm of mobile robot in hallway using PC-camera. International Conference on Automation and Robotics in Construction.

Lowe, D. G. (2004). "Distinctive image features from scale-invariant keypoints." International journal of computer vision **60**(2): 91-110.

Migliore, D., R. Rigamonti, D. Marzorati, M. Matteucci and D. G. Sorrenti Use a single camera for simultaneous localization and mapping with mobile object tracking in dynamic environments.

Mokrzycki, W. and M. Tatol (2011). "COLOUR DIFFERENCE ΔE—A SURVEY." Machine graphics & vision **20**(4): 383-411.

Moravec, H. P. (1988). "Sensor fusion in certainty grids for mobile robots." AI magazine **9**(2): 61.

Murillo, A. C., J. Košecká, J. J. Guerrero and C. Sagüés (2008). "Visual door detection integrating appearance and shape cues." Robotics and Autonomous Systems **56**(6): 512-521.

NOZ-SALINAS, R. M., E. Aguirre, M. García-Silvente and A. G. ALEZ "Door-detection using computer vision and fuzzy logic."

Shi, W. and J. Samarabandu (2006). Investigating the performance of corridor and door detection algorithms in different environments. Information and Automation, 2006. ICIA 2006. International Conference on, IEEE.

Siegwart, R. and I. R. Nourbakhsh (2004). Introduction to Autonomous Mobile Robots, Bradford Company.

Sim, R. and J. J. Little (2006). Autonomous vision-based exploration and mapping using hybrid maps and Rao-Blackwellised particle filters. Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on, IEEE.

Stoeter, S. A., F. Le Mauff and N. P. Papanikolopoulos (2000). Real-time door detection in cluttered environments. Intelligent Control, 2000. Proceedings of the 2000 IEEE International Symposium on, IEEE.

Szeliski, R. (2010). Computer Vision: Algorithms and Applications, Springer.

Thrun, S. and M. Montemerlo (2006). "The graph SLAM algorithm with applications to large-scale mapping of urban structures." The International Journal of Robotics Research **25**(5-6): 403-429.

Tuytelaars, T. and K. Mikolajczyk (2008). "Local invariant feature detectors: a survey." Foundations and Trends® in Computer Graphics and Vision **3**(3): 177-280.

Vasudevan, S., S. Gächter, V. Nguyen and R. Siegwart (2007). "Cognitive maps for mobile robots—an object based approach." Robotics and Autonomous Systems **55**(5): 359-371.

Viola, P. and M. Jones (2001). Rapid object detection using a boosted cascade of simple features. Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on, IEEE.

Zender, H., O. M. Mozos, P. Jensfelt, G.-J. Kruijff and W. Burgard (2008). "Conceptual spatial representations for indoor mobile robots." Robotics and Autonomous Systems **56**(6): 493-502.