

## A METHOD FOR DETERMINING THE POSE OF A HANDHELD WEBCAM

Xianhang Zhang <sup>1</sup>

<sup>1</sup>Dept. of Computer Science, The University of Sydney

**ABSTRACT:** The problem of determining the pose (position and orientation) of a sensor in an unknown environment by incrementally building up a map of the environment is known as Simultaneous Localisation and Mapping (SLAM) in robotics. Currently, the focus for SLAM work has been done on sensors such as laser rangefinders and sonar with very little work on using vision as a sensor. This paper investigates the use of vision in a SLAM system using a hand held, commodity webcam. Experiments show that the implementation could manage to perform SLAM for a moderate period of time but was not completely successful in solving the SLAM problem, mainly due to the relatively small viewing angle of the web camera. More work is called upon to integrate existing classical SLAM algorithms into a visual setting in order to improve accuracy and robustness.

### INTRODUCTION

Computer Vision is the field of computer science devoted to extracting some sort of meaningful description of the world from pictures or sequences of pictures [FP03]. A classic problem of computer vision has been to continually compute the pose (position & orientation) of a moving camera looking onto a scene. This information is useful for applications like adding virtual game characters into a real life scene [ArQuake] or figuring out the position of a robot [GS04] among many other things. Current approaches to solving this problem have arisen independently in 3 different fields and have differed drastically in their approach based on the structure of the problem to be solved.

In the field of Computer Vision and Structure From Motion<sup>1</sup> (SFM), this problem has been called Camera Pose Estimation or Camera Egomotion. In the field of Robotics, this problem is called Simultaneous Localisation and Mapping (SLAM) and in the field of Augmented Reality<sup>2</sup> (AR), this problem is known as Registration.

Although the problem looks similar on the surface in all 3 fields, the reason why techniques developed in one area do not cross over into another is because of subtly different requirements of each approach (See Table 1 for a summary).

Table 1. A summary of the requirements of different approaches

Field	Vision Based	Real-Time	Natural Features
Structure From Motion	Yes	<b>No</b>	Yes
Robotics	<b>No</b>	Yes	Yes
Augmented Reality	Yes	Yes	<b>No</b>
3D Visual SLAM	Yes	Yes	Yes

Structure From Motion has classically concerned itself with computing the camera pose from a sequence of images in which the entire sequence is captured before any pose information is computed. In Robotics, algorithms must be run in real time, that is, the pose must be computed incrementally based on constant new data. However, most robotics algorithms do not use vision as a

<sup>1</sup> Structure From Motion attempts to extract meaningful information from a scene based on the apparent motion of objects between frames.

<sup>2</sup> Augmented Reality uses a headset to overlay objects and information from a virtual world on top of real world scene.

sensor and those that do restrict the robot to lie on a flat plane [DK01, GS04]. Augmented reality systems must also run in real time, however, instead of finding natural features in the scene, known markers are placed in the scene which can be easily detected and tracked.

It has only been recently that researchers have looked into building systems which can run in real time using natural landmarks in full 3D [DGK04]. In this paper, this approach will be termed Visual SLAM. Being able to solve this problem without the previously mentioned limitations represents an important step in Computer Vision because it lets us use cameras in situations that require immediate feedback in an uncontrolled environment. Current Augmented Reality systems tend to be tedious and expensive to set up because the position of the artificial landmarks must be computed very accurately. They can also not be used in environments in which constant change is happening because the markers must be reinitialised every time a change is made. By using natural features, the ease of creating AR systems can be increased dramatically. Robotics systems would also benefit from vision as many of the cues in our environment are visual and vision based sensors are drastically cheaper than laser rangefinders and sonar sensors which are used conventionally in robotics. Visual SLAM systems would represent an important step in building more capable robots.

The main difficulty of Visual SLAM is that landmarks in the scene must be found and initialised in real time and then consequently used to estimate camera position. If selection is not done properly, error in estimation of the landmarks makes the estimation of camera position uncertain which subsequently feedback and make further landmark positions even more uncertain. It is this propagation of error that has thus far hindered research into performing Visual SLAM in real-time.

The aim of this paper is to present an investigation into the requirements of such a Visual SLAM system and some problems involved in building such a system. In particular, the problem of propagation of uncertainty is discussed with a view towards how classical vision algorithms might be adapted to be more suited for real-time work. The existing methods for controlling the propagation of error were found to be inadequate when applied to the domain of vision and a new scheme is proposed that aims to address these issues. Although this new scheme did not fully fulfil the design goals of the system, they did achieve a moderate degree of success and indicate that there such an approach holds potential.

The paper is structured as follows: design goals of the system are listed. Then some background is presented on why this problem remains difficult as well as a brief description of a modern algorithm used in robotics called EKF and why it is non-trivial to adapt to Visual SLAM. A brief description is given of my implementation and why it avoids many of the problems presented by EKF. Finally, some results are given of the system in action followed by conclusions.

## DESIGN GOALS

The goals of for the design of the Visual SLAM system are:

- The system must operate in real time with a frame rate of at least 10 frames per second (ideally 30 frames per second)
- The system must be able to run on a off-the-shelf web camera using commodity PC hardware
- The system will be designed to work only in an enclosed environment such as a room with no significant occlusions. It is not expected for this algorithm to handle outdoor environments. Outdoor environments present a difficult challenge for real-time algorithms since depth information for faraway objects is difficult to estimate in real-time.
- The system will maintain roughly the same level of accuracy, regardless of how long it has been running.

## BACKGROUND

One of the constraints on the Visual SLAM system is that it must work in real time. Although it may seem that this merely means that the algorithm has to be fast enough, there are actually several more subtle constraints on the system which limit the choice of algorithms possible.

First of all, both the running time and memory requirements of the algorithm must remain constant, even during extended protracted runs with many features stored. This means that not all the information gathered by the system can be kept indefinitely. Secondly, such algorithms need to be able to update its estimates incrementally, relying only on past knowledge to form an estimate.

The combination of using natural features and a real-time algorithm means that the propagation of uncertainty and error becomes a critical consideration. Offline algorithms have the luxury of being able to perform global error correction which can limit the amount of error to some upper bound. In Augmented Reality systems in which the features are pre-defined ahead of time, the accuracy of the features can be measured very accurately beforehand. However, with a combination of both real time operation and natural features, the only way to establish the position of *features* is by using inherently uncertain camera data and the only way to establish the position of the *camera* is to use inherently uncertain feature positions. If the system is not designed to explicitly handle this uncertainty, then the accuracy of the system will degrade unchecked over time until the system becomes overwhelmed with error and is useless. This makes both classic Structure From Motion and Augmented Reality algorithms unsuitable for this application.

Robotics algorithms are designed from the start to handle such uncertainty, which makes it seem that trying to use lessons learnt from SLAM in this new domain is the best solution. However, care needs to be taken when trying to adapt such algorithms because of fundamentally different assumptions inherent in different types of sensors. One popular approach known as Extended Kalman Filters (EKF) [MTKW02] is presented below along with some critiques as to why it is unsuited for Visual SLAM. Although alternative algorithms do exist, similar criticisms apply since they stem from fundamental differences.

### EXTENDED KALMAN FILTERS(EKF)

Most classical robotics systems use an algorithm called the Extended Kalman Filter (EKF) to model uncertainty in the system. Briefly, the EKF algorithm models both the sensor and landmarks as a 3D Gaussian probability density function. During each measurement step, the positions and uncertainties of the landmarks and the sensors are estimated and the model is updated to reflect this. The EKF presents a probabilistic way to incorporate and model uncertain measurements into a real-time system. While EKF is popular when used with traditional sonar and laser based robotics systems, there are a number of differences when trying to adapt such algorithms for a vision based system. Visual SLAM differs fundamentally from Laser Rangefinders and Sonar sensors in 3 crucial ways.

First of all, Laser sensors return depth information while vision does not. Instead, vision only tells us that a point lies on a ray but not how far away it is from the camera. Secondly, Vision sensors return a sparse rather than dense set. That is, only a few points in any image are considered relevant for later

processing. On the other hand, Sonar and Laser returns relevant data for every point scanned. Finally, the points returned by Sonar and Laser are anonymous; a point in a scan cannot be distinguished from any other point. On the other hand, points from vision can be distinguished from each other based on appearance.

All of these changes render some of the fundamental assumptions of EKF unsuitable for the vision problem. The EKF assumes that each measurement is *independent* of all other measurements and can be applied in an incremental manner. However, measurements in vision are highly dependant. Because each vision measurement returns a ray rather than a distance, the camera pose can only be computed by the intersection of multiple rays. This means that each update cannot be incremental like the EKF assumes but must be done in a batch since every measurement depends on every other measurement from the same frame to determine the pose. In addition, estimates of the pose are not Gaussian distributed like the EKF assumes either. Pose estimates have 6 degrees of freedom (3 for translation, 3 for rotation) so they are a probability distribution function in 6 dimensional space. A typical pose estimate looks something closer to a hyperboloid in 6 dimensions than a Gaussian function. Because of this, the Gaussian assumption that underlies EKF is not a close fit to the problem of Visual SLAM. Despite this, attempts by [DGK04] have been made to attempt Visual SLAM using an EKF framework which has show moderate success.

Alternatives to EKF do exist like FastSLAM [MTKW02] and or Particle Filters [FTBD00] but these were not investigated. Instead, this paper attempts to presents a new way to estimate uncertainty that tries to take into account some of these differences.

## IMPLICATIONS

The main implication of moving to a vision system is that we are now dealing with rays rather than points. If we imagine a standard perspective camera like that depicted in Figure 1, then all we can say about a landmark from one view is that it lies on the ray passing from the camera centre through that point on the image plane. If we want to establish the actual position of a landmark, then we need to observe that landmark through numerous camera views at numerous angles and figure out where the rays intersect.

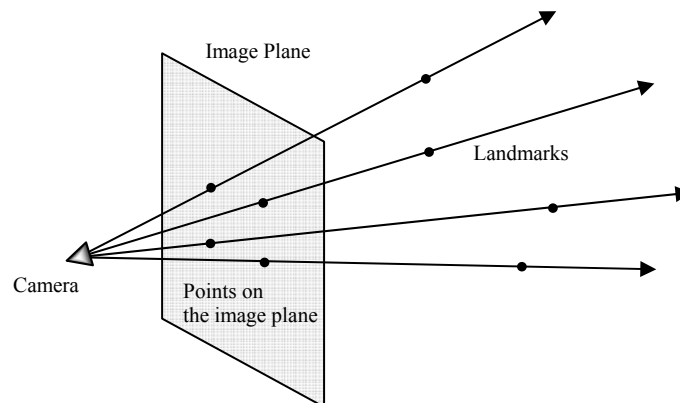


Figure 1. The standard perspective camera is comprised of a camera centre and an image plane. A line projected from a landmark in the scene to the camera centre will pass through the corresponding point on the image plane.

Similarly, figuring out the pose from a single landmark is impossible. Instead, the intersection of rays back-projected from multiple landmarks is needed to compute the pose (Figure 2). At minimum, 3 landmarks are required to compute a unique pose but more landmarks can be used to improve accuracy and combat noise.

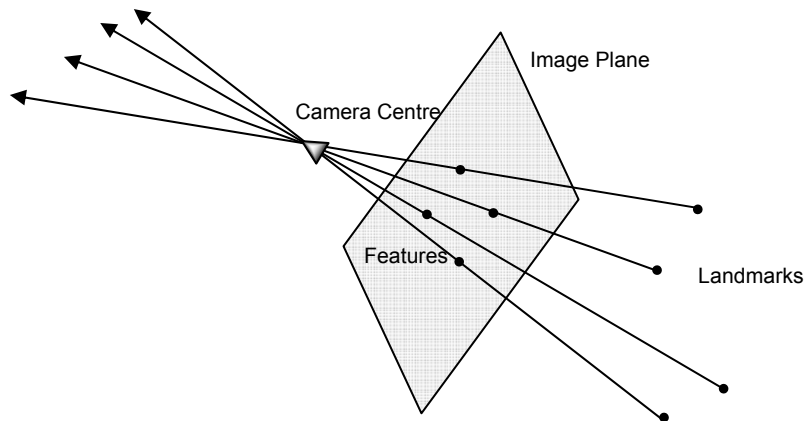


Figure 2. If the positions of enough landmarks are known, then it is possible to figure out the pose of the camera using a similar principle.

### SYSTEM SETUP

The system is designed to use corner points in an image as landmarks. A modified Shi-Tomasi corner detector and tracker [Shi94] was used which also took into account the surface normals and distance from the camera when computing correlation. This was done because corner points are easy to find and track and can usually be easily distinguished from each other. The system starts out looking at a *bootstrap pattern* of 8 corner points whose location is known beforehand (See Figure 3). Because these 8 landmarks are known, the camera pose can be computed using existing Augmented Reality style algorithms. Once the camera pose is known, newer, natural features based on corners can then be found in the scene. After a suitable number of observations of a certain feature from a wide variety of views, its position can be established accurately enough such that it can be used for further pose estimation. When enough natural features are found, then the features from the bootstrap pattern are no longer necessary and the system can proceed by iteratively finding landmarks, establishing landmarks and then using the previously established landmarks to compute camera pose.

While this sounds simple in abstract, there are a number of details which make it difficult to implement such that uncertainty is minimised. One major issue is when a landmark is deemed reliable enough to use for pose estimation. If a landmark is viewed from a wide range of angles, then the estimate of its position will be more certain. However, delaying the establishment of landmarks also means that fewer landmarks are available in any given view for use in pose estimation. This makes the pose estimation step less certain. What is needed is a method for establishing landmarks in an incremental and reliable manner such that there are both a sufficient number of them and they are sufficiently free from error so as to minimise the error when estimating pose.

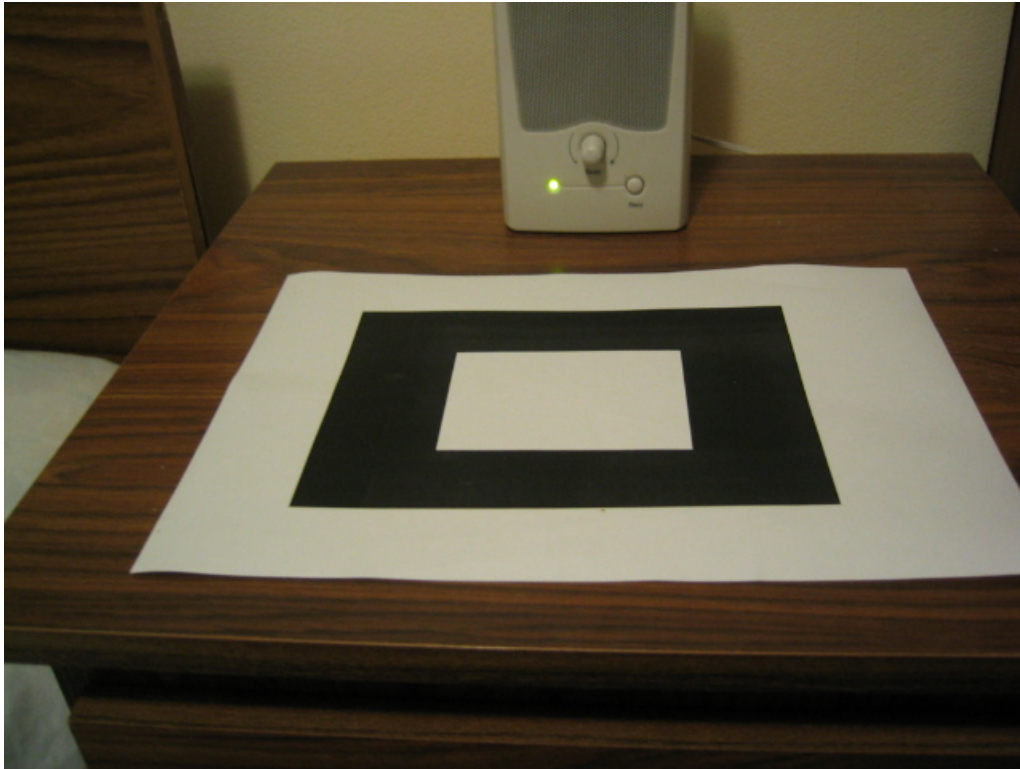


Figure 3. A typical view by the camera with the bootstrap image in the frame. The 8 corners of the image are known to the algorithm beforehand.

The fundamental heuristics I am using for controlling noise is that landmarks initialised earlier in the map are likely to be more accurate than points initialized later in the map and that when multiple observations agree on roughly the same value, then it is more likely that they are all correct rather than they are all wrong.

Every landmark and every landmark ray has a reliability number attached to it. The initial features from the bootstrap pattern are initialized with a reliability number of 0 indicating that they are the most reliable features.

During pose estimation, a pose can either be consistent or inconsistent. A consistent pose is one in which all the landmarks used all agree with each other about the pose to some desired accuracy level. An inconsistent pose is one in which at least one landmark disagrees. If a pose is detected to be inconsistent, then we can find the landmark with the highest reliability number (ie: the most unreliable) that disagrees with the pose. This landmark is then removed and the pose is computed again until either the pose becomes consistent or there are only 4 rays left in the set. The reliability number of that pose is then assigned to be one larger than the 4th most reliable of the rays that still remain in the set. The reason why the 4th ray is chosen is because 4 rays are needed to define a unique pose. With 3 rays, a perfect pose can always be found, however, with 4 rays, it is possible to never find a consistent pose. So if at least 4 rays agree with each other on the pose, the pose is likely to be as accurate as the least accurate of the 4 rays plus some error inherent in pose estimation.

For all the rays being projected out from that pose during the triangulation step, the reliability number of that ray will be the reliability number of the current pose. To compute the reliability of the landmark, triangulation is performed based on the rays describing the landmark. Again, the triangulation can either be consistent or inconsistent. If the triangulation is inconsistent, a similar method is used to make it consistent by continually removing the least reliable rays that disagree. However, the difference is that for triangulation, rays can continue to be removed until only 1 ray is left. It is impossible to be inconsistent with only 1 ray so the most reliable ray will always remain. If a ray is marked inconsistent a certain number of times, then it is deleted from the list of rays permanently because it is assumed to be an outlier. A landmark can only be used for pose estimation if a certain number of rays all agree on a consistent point. If a consistent solution cannot be found with enough

rays, then the landmark is then marked unreliable and not used for pose estimation in the next frame. The reliability number of the landmark is again, assigned to be one larger than the 4th most reliable of the rays that still remain in the consistent set.

Because every landmark is tested in every instance, a landmark becomes reliable enough simply when it agrees with more reliable landmarks. If two landmarks do not agree, then the most reliable one is chosen. Furthermore, this algorithm makes no assumptions about the probability distribution of poses. For each instance, the pose is recomputed so it is not necessary to assume a simplified probability function. This avoids the two key weaknesses of the EKF algorithm.

For example, assume we have a camera looking on landmarks with reliability numbers of {3, 5, 6, 8, 2, 0} (Figure 6). Of these, the landmark with a reliability number of 6 does not agree with all of the other landmarks about the camera pose and is removed. When the pose is recomputed, all the landmarks now agree with each other so no further pruning is required. Based on the rays that are left, the ray with a reliability number of 5 is the 4<sup>th</sup> most reliable so the pose has a reliability one greater which is 6. Now, all the landmarks the camera observed in this pose have a ray added with a reliability number of 6 and the process is repeated to compute the position of each landmark.

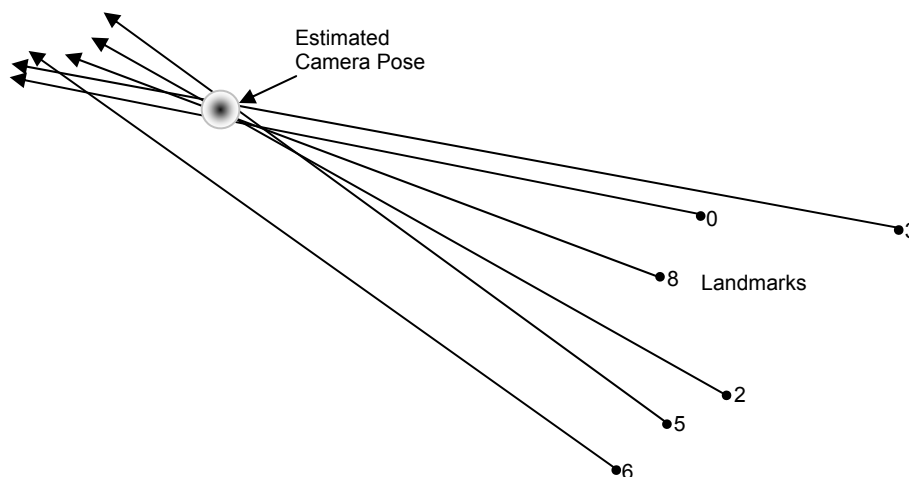


Figure 4. Each ray is marked with a reliability number. All of the rays except ray 6 agree that the camera is located at the estimated position. Because the 4<sup>th</sup> most reliable ray has a reliability number of 5, this pose estimate has a reliability of 6.

## RESULTS:

Due to the nature of this research, it has proven difficult to get any sort of “objective” results about the system as a whole. Because there is no way of determining the true pose of the camera, any evaluation must be by nature subjective. It is also hard to judge this system against similar ones since they all require different assumptions to hold which could not be replicated. Thus, by necessity, much of the results are subjective in nature and depend on a subjective evaluation of how well the system kept with the design goals rather than any objective measure of absolute error terms.

Based on observation, the system was able to maintain a high level of accuracy during the initial stages in which the bootstrap image was present. Once the bootstrap image was removed, the system still was able to reliably find and establish landmarks but there was a noticeable amount of distortion in the positions. After several minutes of running, the error became so bad that pose estimates were effectively useless. Returning the camera back to the bootstrap image constantly allowed the estimates to be stable for longer but the system was not able to keep its accuracy from degrading without the use of prior known landmarks. If the camera strays too far from the bootstrap image, then the error will be so bad that even if it does eventually return back, it will be unable to find the bootstrap image because it thinks it is somewhere else.

The main reason that accuracy degraded seemed to be due to the low field of view of the camera used. Tests on a single point showed that the wider the range of angles a point is viewed from, the more accurate it becomes (Figure 5). Due to the low field of view, landmarks could only be observed from a relatively small range of angles at any one time. This limits the amount of times that each landmark can be seen before it must be established and used. Because of this, when the bootstrap image goes out of view, the new features that have been detected are still relatively inaccurate, about  $\pm 2$  cm judging from their appearance on the map. Because these new features are inaccurate, any features that they then establish are even more inaccurate which leads to error rapidly accumulating. The use of a camera with a large viewing angle should greatly increase the accuracy of the system since it would allow more features to be seen at any one time and a better view of each feature.

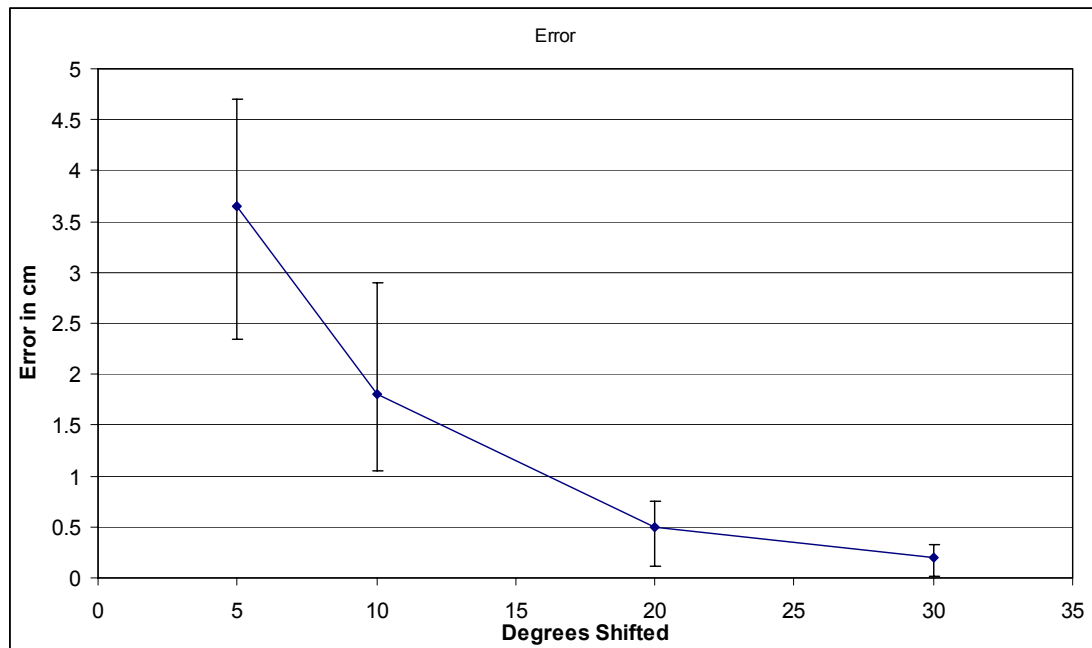


Figure 5. The accuracy of triangulation based on the number of degrees shifted. Accuracy increases markedly at around 20 degrees.

The performance of the system was adequate based on the original design goals. All images were taken at a resolution of  $320 \times 240$  pixels. Running a performance analysis over the algorithm as it currently stands, the times for each operation break down as follows:

Table 2. A breakdown of the runtime performance of Visual SLAM

Step	Time Taken
Feature Detection	20 – 30ms
Feature Tracking	10 – 15ms
Pose Estimation with outlier removal	5 – 10ms
Triangulation	>1ms
Re-detecting old features	>1ms

As can be seen from Table 2, in the worst case, the entire algorithm takes approximately 60ms per frame. This gives a runtime of around 16 frames per second which is considered acceptable but not ideal based on the original design goals. However, it is unnecessary to run feature detection over every single frame. Instead, it is possible to only run feature detection occasionally, for example, when too many features go out of view. If we ran the feature detector only once every 4 frames, then the worst case performance is 30ms which is considered ideal under the design goals.

## CONCLUSIONS

An integrated Visual SLAM system is presented using a single handheld camera, running off a conventional desktop computer. Unlike many previous algorithms, this system runs in real time, uses natural features and works in the full 3 dimensions with 6 degrees of freedom. The key challenge in building the system has been to estimate and control the propagation of errors that is inherent in any system that runs in real time using natural features.

During the course of building the system, a new method was developed to control for the propagation of errors. Although the system as a whole did not perform well enough to fulfil the original goals of the project, they still show a lot of promise and many avenues have been left unexplored that may yield substantial improvements in accuracy. Due to the relative lack of investigation into the application of robotics SLAM techniques as applied to the field of 3D vision, the methods proposed were fairly ad-hoc and it is almost certain that more research into this field will yield much more sophisticated algorithms that should decrease the error rate significantly. Furthermore, it is unclear exactly how much of the error is due to limitations in the hardware used. A camera with a wider field of view is likely to significantly increase the quality of the triangulation phase since a landmark can be viewed from a much wider range of angles and this currently appears to be the largest source of error. The use of stereo cameras would eliminate the need for this error prone triangulation step which should also increase the accuracy of the algorithm significantly. Overall, this investigation represents a preliminary foray into the field of Visual SLAM systems and indicates that there is still much work to be done.

## REFERENCES

- [ARQuake] B. Thomas, B. Close, J. Donoghue, J. Squires, P. De Bondi, M. Morris, and W. Piekarski. ARQuake: An Outdoor/Indoor Augmented Reality First Person Application, *4th International Symposium on Wearable Computers*, pp 139-146, Oct 2000.
- [DK01] A. Davison and N. Kita. 3D Simultaneous Localisation and Map-Building Using Active Vision for a Robot Moving on Undulating Terrain, *Proc. IEEE Conference on Computer Vision and Pattern Recognition, Kauai*. 2001
- [DGK04] A. Davison, Y. Gonzales, N. Kita. Real-Time 3D SLAM with Wide-Angle Vision, *Proc. IFAC Symposium on Intelligent Autonomous Vehicles*. 2004
- [GS04] M. Garcia & A. Solanas. 3D Simultaneous Localization and Modeling from Stereo Vision. *Proceedings of the 2004 IEEE International Conference on Robotics & Automation, New Orleans, LA*. April 2004
- [FP03] D. Forsyth, J. Ponce, Computer Vision, A modern approach., Prentice Hall, ISBN 0-13-191193-7.
- [FTBD00] D. Fox, S. Thrun, W. Burgard, and F. Dellaert. Particle filters for mobile robot localization. In *Sequential Monte Carlo Methods in Practice*. 2000.
- [MTKW02] M. Montemerlo, S. Thrun, S., D. Koller. and B. Wegbreit. FastSLAM: a factored solution to the simultaneous localization and mapping problem. *Eighteenth National Conference on Artificial Intelligence* 593-598, 2002.
- [ST94] J. Shi, C. Tomasi, Good Features to Track, *IEEE Conference on Vision and Pattern Recognition*, 1994.