Markerless Augmented Reality for Robots in Unprepared Environments

Ian Y-H Chen and **Bruce MacDonald** and **Burkhard Wünsche**
University of Auckland, New Zealand
{i.chen, b.macdonald}@auckland.ac.nz, burkhard@cs.auckland.ac.nz

Abstract

Augmented Reality (AR) can assist humans in understanding complex robot information, and improve Human and Robot Interaction (HRI). However, many restrictions are imposed by the underlying technology used and thus have limited current AR systems to operate in controlled or modified robot environments. This hinders the wide spread use of AR for different robot applications. This paper presents a markerless AR system that combines recent tracking and detection techniques for AR visualisation of robot task relevant information. We employ natural feature tracking techniques to compute the camera pose for accurate registration of virtual objects. Automatic relocalisation of the camera pose is achieved using a planar object detection algorithm which recovers from tracking failures. Experiments using a camera mounted on a mobile ground robot demonstrated accurate tracking and successful recovery of planar features in an unprepared indoor environment.

1 Introduction

Technological advancement has made it possible to install a variety of sensors onto a robot platform. Data produced by these sensors typically come in large quantities and high dimensions which are difficult for humans to digest in a short period of time. A separate display of robot information is commonly presented to the users, however, it places cognitive load on the users as they are required to mentally relate large disparate sets of information. AR presents an alternative approach to the visualisation of complex information in real time. Real and virtual data can be visualised in a single, coherent display. Many researchers have seen the benefits AR visualisation and brought such techniques into various fields of robotics. AR assists humans to understand robot behaviour and interpret robot data in an intuitive manner. Robot information, such as sensor, map, and task relevant data, can be overlaid in context with the real world environment. This effectively maximises the shared perceptual space between the user and the robot [Collett and MacDonald, 2006]. Application of AR in robotics has been shown to increase situation awareness in robot teleoperation, assist robot development, and improve Human-Robot Interaction (HRI), as will be reviewed in Section 2.

Typically marker based tracking methods are used to track objects to be augmented in an AR scene. Although these methods are a fast and low cost solution for creating AR, partial occlusion of the markers or direct exposure to strong lighting conditions may cause tracking to fail [Azuma, 1999]. Modifying the environment to provide markers is particularly challenging in mobile outdoor environments. Tracking of natural features can overcome these limitations and scale AR to operate in unprepared environments. This paper describes a markerless AR system based on tracking of planar features in the environment. We combine feature tracking and object detection algorithms to create AR that is able to recover from erratic motions and occlusions of the camera. Real time performance is achieved through a multithreaded system implementation. We apply our markerless AR system to assist robot development by enabling users to arbitrarily introduce virtual objects into the robot environment for creating test scenarios. A camera mounted on the robot is used to provide real world images on which virtual objects are overlaid. The visual feedback describes the world as seen from the robot's perspective where virtual objects are accurately registered in their corresponding geometric locations.

The remainder of the paper is organised as follows: Section 2 describes related work, Section 3 explains the problem to be solved, Section 4 details our markerless AR implementation, Section 5 describes the system design, and Section 6 presents results and discusses the limitations.

2 Related Work

This section presents recent markerless AR tracking techniques and gives an overview of AR applications in robotics and the underlying technologies.

2.1 Markerless AR Tracking Techniques

Many markerless tracking techniques have emerged recently to facilitate AR in unprepared environments. A number of these AR systems rely on point feature tracking algorithms. In particular, points extracted from planes in the environment are commonly used for computing the camera pose [Lourakis and Argyros, 2004; Simon and Berger, 2002]. They have been demonstrated to operate in various environments, including outdoors. However, the drawback inherent in point feature tracking is that points are easily lost due to sudden camera motions and occlusions.

Line or edge tracking can be used to overcome the limitations in point tracking systems as lines are less susceptible to partial occlusions. Line tracking has been applied in model-based AR systems to achieve robust and accurate AR registration results [Comport *et al.*, 2004; Klein and Murray, 2006], but it places a burden on users to build models of existing objects in the environment prior to tracking.

An alternative approach for improving the reliability of feature tracking is to use highly distinctive feature descriptors, such as in [Skrypnyk and Lowe, 2004; Mikolajczyk *et al.*, 2005]. These region based descriptors enable wide-baseline matching of features that is invariant to scale and rotation. This technique also allows recognition of 3D objects as well as previously visited scenes. The downfall normally lies in expensive computational requirements during feature extraction, which presents a barrier for these AR systems to perform at interactive frame rates.

Another solution for localising a free moving camera is to apply visual SLAM [Davison *et al.*, 2007; Klein and Murray, 2007]. The camera is tracked while a map of the environment is constructed. Features, such as points, edges, or descriptors are continuously extracted as the camera moves around the environment, and the features' world positions are iteratively refined. As the map size grows, the time needed for SLAM updates increases. Thus, the scalability of visual SLAM AR systems is dependent on the processing power of the host computer.

2.2 AR in Robotics

AR has long been used to increase the robot operator's situation awareness by providing a global view of the robot environment. Recently, Sugimoto *et al.* [2005] use AR to assist remote users in robot teleoperation by synthesizing a virtual robot onto the centre of real world images captured from the robot's onboard camera. This helps the operators to understand the spatial relationship between the robot and the environment. The captured images are continuously stored in a database and a selection algorithm picks the most suitable image on which the virtual robot is overlaid.

Young *et al.* [2006] use AR to improve HRI by communicating robot information through the use of bubblegrams, which contain robot status to be visualised to users wearing Head Mounted Displays (HMD). Haar-like features [Lienhart and Maydt, 2002] are used to detect the target robot in the input video image and 2D virtual bubblegrams are superimposed at a location near the target robot. No 3D virtual information is introduced, therefore, the pose of the user's viewing direction is not tracked in this case. Dragone *et al.* [2007] also use AR to improve HRI by placing virtual characters on top of real robots to help express robot states through natural interaction modalities such as emotions and gestures. The user interacts with the robot by wearing a HMD and the user's viewpoint is obtained by tracking ARToolkit [Kato and Billinghurst, 1999] markers attached to the robots.

Collett and MacDonald [2006] apply AR for debugging robot applications. They overlay virtual information such as robot sensory and internal algorithm state data onto real world images of the robot environment to provide robot developers better understanding of the robot's world view. A fixed overhead camera is used to track ARToolkitPlus [Wagner and Schmalstieg, 2007] markers attached to the ground robots in the test environment. The pose of each robot can then be computed and the virtual information is accurately overlaid. A similar technique is used by Kozlov *et al.* [2007] to help robot developers debug Simultaneous Mapping and Localisation (SLAM) algorithms. Stilman *et al.* [2005] also create an AR environment for decoupled testing of robot subsystems. Results from motion planning and vision algorithms are visualised over real world images provided by both external cameras and a camera mounted on the robot. A set of markers is associated to each physical object in the environment and the markers are tracked by multiple external cameras to compute the pose of the objects.

The literature reveals that most of the above AR applications in robotics rely on markers for recovering the camera position and orientation. In comparison, we recently propose a markerless AR system [Chen *et al.*, 2008] that tracks the pose of a camera mounted on a robotic helicopter using natural features. Under a known initial configuration between the camera and the ground plane, a virtual marker is inserted into the scene. The position of the virtual marker on the image plane can be continuously updated by tracking natural features in the environment. The four virtual marker vertices are used to compute the camera pose in the same manner as marker-based AR systems. The technique is highly dependent on the number and quality of natural features being tracked, and AR visualisation can no longer proceed when an insufficient number of features is tracked. Accurate tracking of translational camera motions is achieved but performs poorly under certain camera rotational motions in presence of noisy features.

This paper improves our previous method for markerless AR tracking, by applying a combination of state-of-art tracking and detection algorithms to achieve AR that allows tracking failures to be automatically recovered for improved system robustness. The method for initialisation of tracking is modified, enabling a user to select the initial tracking points on a plane. We focus the tracking on the user specified planar region for determining the camera pose. Another major improvement presented is the integration of an online learning and detection algorithm for recovering from tracking failures.

3 Problem Description

As we aim to apply AR for robots in unprepared environments, it is assumed that an external camera overlooking the robot environment is not available. This commonly occurs when executing robot tasks in outdoor environments. An example is the development of aerial robot applications where a global camera view of the experimental environment can not be obtained, and HMD's may not be appropriate since the aerial robot would not always be in the view of the human user. In this research, we investigate AR visualisation using a single camera mounted on a moving robot. This can also help robot development in a number of ways: realistic virtual objects can be overlaid to test image processing algorithms, obstacles can be introduced to evaluate navigation tasks, and waypoints can be added to guide robot teleoperation.

We assume that the target environment contains some planar structures. This is a reasonable simplification of the problem, since many applications of robots are in urban or rural environments and man-made structures tend to have planar surfaces. The intrinsic camera parameters are assumed to be known and accurately calibrated beforehand.

The problem to tackle is the tracking of the camera pose as the robot moves around the environment. With accurate estimates of the physical camera position and orientation, the pose of the virtual camera can be updated correspondingly. Any virtual objects introduced should be accurately registered and remain aligned with the live background video images, appearing as if they are part of the physical environment.

Natural features in the environment are to be tracked instead of markers to compute the camera pose. In an ideal situation, all features to be tracked should remain within the camera view throughout the whole video sequence. However, this does not always happen in practice. During tracking, features are easily lost due to sudden rapid camera motions, occlusions, motion blur, or changes in illumination in the environment. When tracking fails, virtual objects would become incorrectly registered or completely lost. Therefore, there is a need to relocalise the camera whenever tracking failure occurs. The process should be automatic and avoid any manual user interventions.

4 Methodology

We combine tracking and detection in the design of our AR system for improved robustness and camera relocalisation capability. We first explain how the camera pose can be obtained by tracking natural features in the scene. In particular, points that lie on a plane are tracked. Recovery of tracking failure using a planar object detection algorithm is then presented.

4.1 Feature Tracking for Camera Pose Estimation

Four co-planar points from the scene are required for tracking in our markerless AR system. These are selected by the users as they manually choose four 2D points on the input image that corresponds to a planar region in the 3D world. Interest points are extracted from the input positions and tracked using the Kanade-Lucas-Tomasi (KLT) feature tracker [Tomasi and Kanade, 1991; Shi and Tomasi, 1994]. We will refer to the tracked co-planar points as KLT points. The point selection process is performed by the user online and the camera does not need to be stationary. Tracking of each feature point will immediately start as soon as it is selected.

The four points selected on the 2D image plane are mapped to a rectangle on a planar surface in the 3D world to obtain their 3D world coordinates using a method presented in [Simon *et al.*, 2000]. Given that the 3D world coordinates of the KLT points are now known and intrinsic camera calibration information is also available, the camera translation and rotation parameters can be computed using the pose estimation algorithm given by [Dementhon and Davis, 1995]. AR can now commence with virtual objects projected onto the view of the camera, however, it operates in a space with ambiguous scale. Most often when performing AR in robotics, it is necessary to upgrade the results to a Euclidean space that corresponds to the scale of the real world in order to overlay virtual objects, such as range sensor readings, onto the scene with correct real world dimensions. To do this, we choose to pre-measure the length of an edge of the planar target to be tracked for determining the real world scale.

For more robust computation of the camera pose, Virtual Visual Servoing (VVS) [Marchand and Spindler, 2005] is used to minimise the error in the pose estimates, which helps to reduce jitter.

4.2 Planar Object Detection

To recover from tracking failure, a fast, scale and perspective invariant planar object detection algorithm named "Ferns" [Özuysal *et al.*, 2007], is integrated to enable real time recognition of previously selected (trained) objects. The Ferns detector treats object detection as a classification task. Ferns takes a Semi-Naive Bayesian classification approach for recognising features from input images, and the goal is to classify a given image patch during the detection phase into the most likely class using simple image tests.

A training phase is required before the online detection process and is typically performed offline. Keypoints are first extracted from the target planar object and image patches around the keypoints are warped based on random affine deformations to generate many possible appearances of the planar object as seen from different viewpoints for training. The set of all different views of an image patch around a keypoint results in a class. A large number of binary features are required during the training phase to construct the posterior distributions needed for classification. Each binary feature simply describes the difference between the intensities of two randomly selected pixel locations within an image patch. These binary features are divided into groups, known as ferns, and during the detection phase, their outputs are combined to calculate the probability of a given image patch belonging to one of previously trained classes.

Not all input keypoints need to be matched to those extracted during the training phase for a successful classification. Given a sufficient number of matches with high confidence, the planar object can be detected. This strengthens the system against partial occlusions.

In our implementation, we deploy the training of the classifier online in a background thread while KLT tracking takes place. Immediately after the four KLT points are specified, the input image of the scene is stored. The rectangle formed by the KLT points defines the region of interest for training using Ferns. Once the training is complete, planar object detection begins and is used to correct and recover any KLT points that are lost during tracking.

4.3 Algorithm Overview

To put the algorithm into perspective, the flow of operations is now described. To initialise the AR system, the four co-planar points selected by the user are tracked

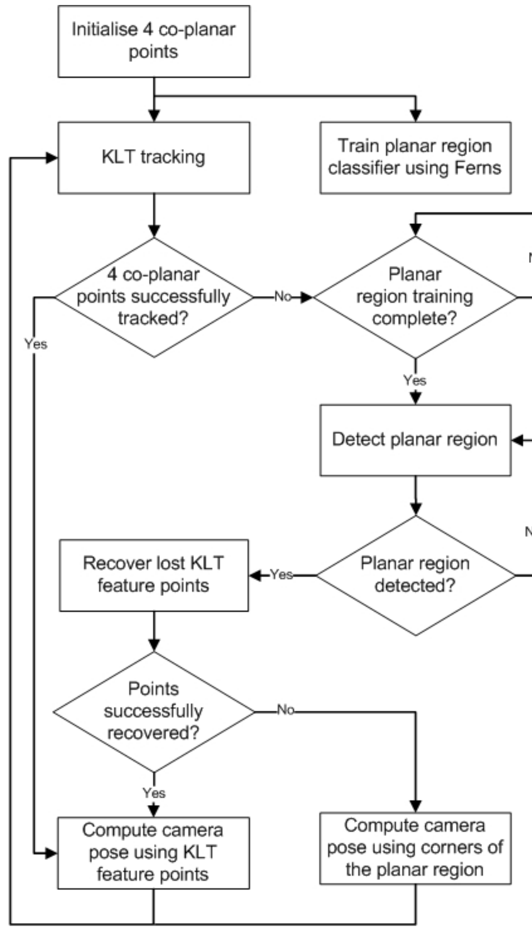


Figure 1: Decision flow of the markerless tracking and detection algorithm

using KLT. The region formed by the four KLT points is stored and used to train a planar object classifier in the second thread using Ferns. While training takes places, the augmentation of virtual objects also begins using the camera pose computed from the KLT points. As soon as the training is complete, the Ferns detector executes periodically to correct the positions of the tracked KLT points.

At any time a KLT point is lost, the algorithm checks to see whether the corresponding vertex of the planar region detected by the Ferns detector is still within the view of the camera. The KLT point is automatically reselected if the vertex is visible; otherwise, the vertices of the detected planar region are used as substitutes for the computation of the camera pose to allow AR visualisation to continue. Fig. 1 gives a flow chart of the overall process.

5 System Design

In our implementation, we separate tracking and detection in two threads that execute in parallel. This allows

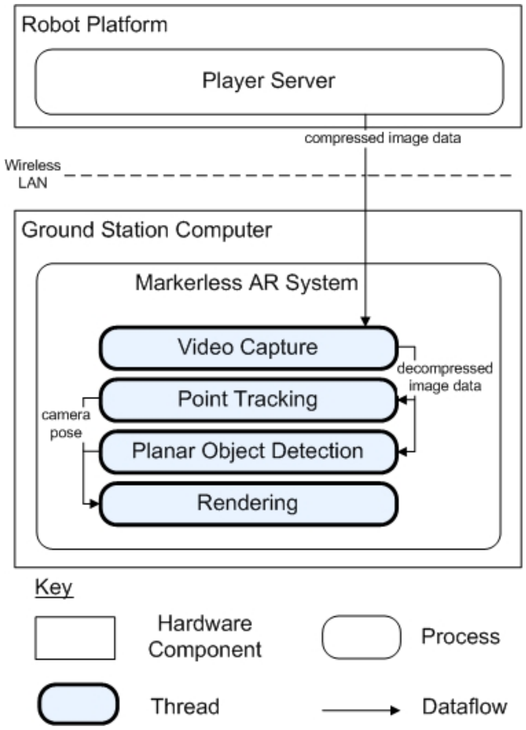


Figure 2: Multi-threaded system design and dataflow

augmentation of virtual objects to proceed over smooth live video sequences while expensive training and detection operations take place in the background. Real time AR is therefore achieved. Fig. 2 illustrates our multi-threaded system design.

Player Server [Player/Stage, 2008] is a robot development software tool that provides an interface to underlying hardware on the robot platform. Compressed image data from the camera sensor is transmitted through Player to the local machine over wireless LAN. The image data is then decompressed and used for tracking and planar object detection. Once the camera pose is computed, visual augmentation can take place by rendering the scene from the estimated camera viewpoint using a 3D graphics rendering engine OGRE [2008].

6 Results and Discussions

An experiment investigated the computation times required by the tracking and the detection components of the proposed algorithm. A handheld camera was used to detect and track a target planar object for a period of 30 seconds. The mean and standard deviation of the computation times are presented in Table 1. The system was deployed in Linux on a 2.4GHz Intel(R) Core(TM)2 Quad CPU with 1GB of RAM and a NVIDIA Quadro FX 3450/4000 SDI graphics card. A Logitech Quickcam Fusion camera was used to capture live video images in resolution of 640x480.

KLT Tracking	5 ± 1ms
Ferns Detection	28 ± 4ms

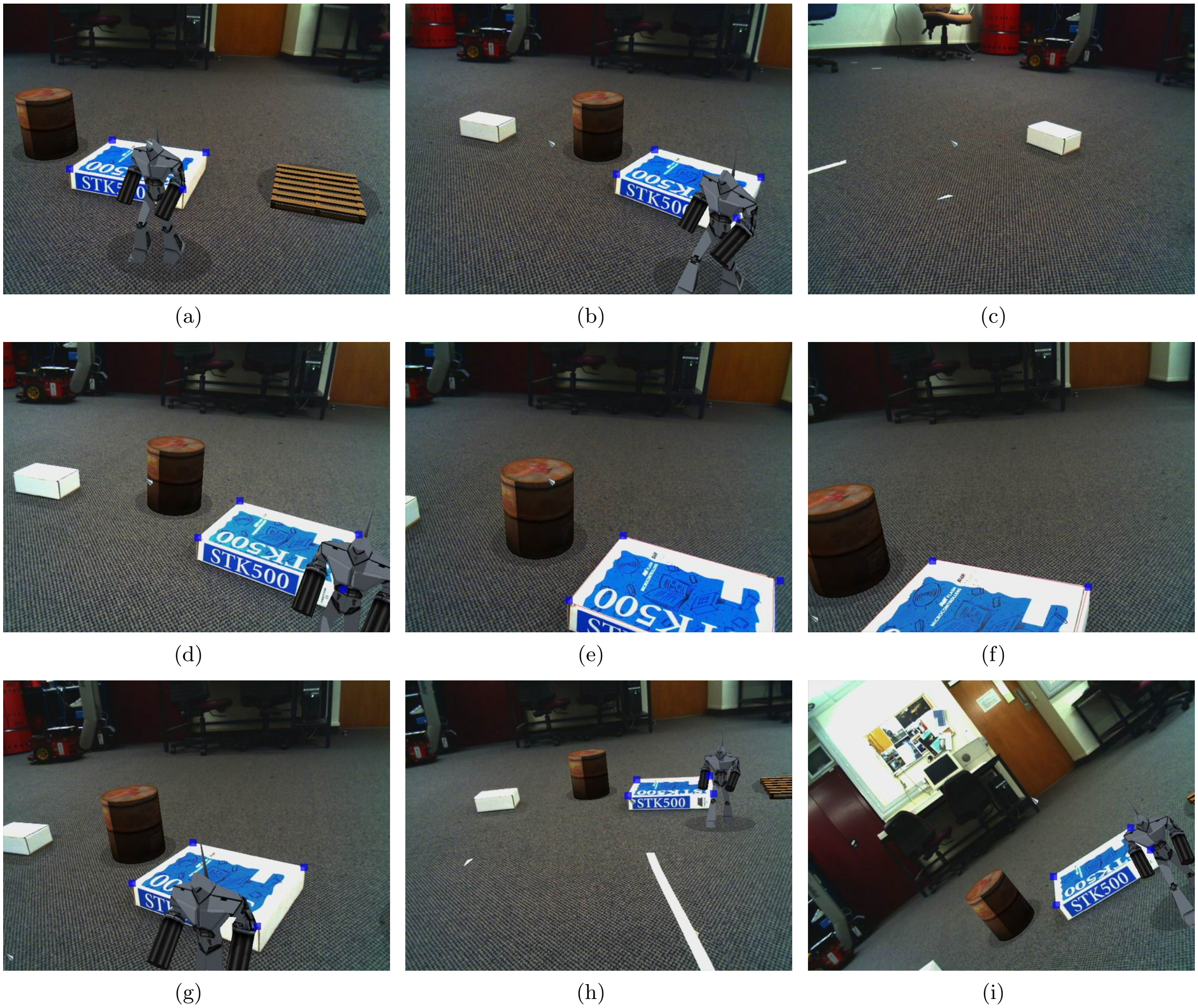
Table 1: Computation time of each algorithm component

It is important to note that the planar object training process is expensive and can take a considerable amount of time (up to 1 minute) depending on the size of the image region being trained. However, this is a one-off overhead and the process is transparent to the user since it has no influence on the quality of AR visualisation. A problem is that if KLT points are lost during this time, visual augmentation of virtual objects is temporarily lost until the training is complete. The user can also choose to save the trained classifier data to skip the training phase in later experiments with the same planar object.

We also present results corresponding to a video sequence of AR which took place in an unprepared indoor environment. The same camera was used and mounted on a Pioneer 3DX robot for video capture, see Fig. 3. The robot was manually controlled to move within a radius of 2 metres of the tracked target object. We augmented various virtual objects over the planar target and moved both the camera and the robot around to observe the target environment from different viewpoints, see Fig. 4. The frame rate of this experiment was limited to 9-10 Hz, due to video capture over wireless LAN.

Visual results indicate successful tracking of the planar target within the specified working area and the virtual objects remained accurately registered, with an error between the real and the estimated camera positions measured to be approximately 0.0118 metres. As the robot moved around the target environment, the planar target constantly exited the camera view. The Ferns detector was able to detect the planar target when it reappeared and automatically resumed KLT tracking, e.g. Fig. 4(c),

Figure 3: Pioneer 3DX robot with a camera mounted on the robot arm



(a) (b) (c) (d) (e) (f) (g) (h) (i)

Figure 4: A barrel, an animated character, and a wood pallet are introduced into the scene. (a) Four points (in blue) on a planar surface of a box are manually selected and AR begins while training takes place, (b) robot turning away from the planar target, (c) planar target exits the camera view and augmentation is lost, (d) Planar target immediately detected when reappeared and KLT points are recovered, (e) & (f) Planar target partially occluded and AR continues using the estimated positions of the planar region corners, (g) KLT points are again recovered and tracking is resumed, (h) camera viewing the planar target from a distance, (i) camera viewing the planar target under different rotations.

4(d). Partial occlusion of the planar target was also acceptable with small jitters in the visual augmentation.

The results show significant improvements over the method presented in [Chen *et al.*, 2008]. The new approach allows AR to commence from any starting camera position by letting the user to choose the target plane to be tracked. Tracking is then limited to the selected planar region, which reduces the likelihood of tracking noisy features. Accurate and reliable camera pose estimates are obtained for more effective AR. A static environment is no longer assumed, since automatic detection of the target plane will allow AR to resume after tracking failures from occlusions due to moving objects.

There are a couple of situations that our AR system can fail. This happens when both tracking and detection of the planar region are unsuccessful. The first situation occurs during the training phase of the detection algorithm as mentioned before. The second is when the camera observes the target environment from long distances or at small acute angles between the viewpoint and the planar surface. Erroneous classification results are produced by the Ferns detector, and in some cases, the target object can not be detected. Visual augmentation is lost if KLT tracking also fails during this period.

7 Conclusions and Future work

We have presented a markerless AR system that tracks an onboard robot camera using a combination of tracking and detection algorithms. The user selects four co-planar points to be tracked which are used for the computation of the camera pose. During tracking failures, a planar detection algorithm is applied to recover the lost feature points and resume AR visualisation.

An experiment was performed using a mobile ground robot moving in an unprepared indoor environment. Results show accurate tracking and successful recovery of planar features. The target planar region is correctly recognised when returned to the camera view and tracking can proceed as normal. Nevertheless, the AR visualisation is temporarily lost when both the tracking and detection fail. This normally occurs when the camera exceeds certain viewing angles.

Future work aims to extend the working area of our markerless AR system. Currently, the working area is limited to scenes where the planar region is visible within the camera view. This lowers the usability and scalability of the system for robot development. The system should have the ability to augment virtual objects anywhere the robot moves to. This limitation needs to be addressed in the future in order to apply our system in a wider range of robot applications.

References

[Azuma, 1999] Ronald T. Azuma. *Mixed Reality: Merging Real and Virtual Worlds*, chapter 21, pages 379–390. Springer-Verlag, Berlin, 1999.

[Chen *et al.*, 2008] Ian Yen-Hung Chen, Bruce MacDonald, and Burkhard Wünsche. Markerless augmented reality for robotic helicopter applications. In *Proceedings of the 2nd Workshop on Robot Vision*, Auckland, New Zealand, February 18-20 2008.

[Collett and MacDonald, 2006] T.H.J. Collett and B.A. MacDonald. Augmented reality visualisation for player. In *Proceedings of the 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pages 3954–3959, 2006.

[Comport *et al.*, 2004] A.I. Comport, E. Marchand, and F. Chaumette. Robust model-based tracking for robot vision. In *Intelligent Robots and Systems, 2004. (IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, volume 1, pages 692–697 vol.1, 2004.

[Davison *et al.*, 2007] Andrew J. Davison, Ian D. Reid, Nicholas D. Molton, and Olivier Stasse. MonoSLAM: Real-time single camera SLAM. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(6):1052–1067, 2007.

[Dementhon and Davis, 1995] Daniel F. Dementhon and Larry S. Davis. Model-based object pose in 25 lines of code. *International Journal of Computer Vision*, 15(1):123–141, June 1995.

[Dragone *et al.*, 2007] Mauro Dragone, Thomas Holz, and G.M.P. O'Hare. Using mixed reality agents as social interfaces for robots. In Thomas Holz, editor, *The 16th IEEE International Symposium on Robot and Human interactive Communication, 2007. RO-MAN 2007.*, pages 1161–1166, 2007.

[Kato and Billinghurst, 1999] H. Kato and M. Billinghurst. Marker tracking and hmd calibration for a video-based augmented reality conferencing system. In *Proceedings of the 2nd IEEE and ACM International Workshop on Augmented Reality, 1999. (IWAR '99)*, pages 85–94, 1999.

[Klein and Murray, 2006] Georg Klein and David Murray. Full-3d edge tracking with a particle filter. In *Proceedings of the British Machine Vision Conference (BMVC'06)*, volume 3, pages 1119–1128, Edinburgh, September 2006. BMVA.

[Klein and Murray, 2007] Georg Klein and David Murray. Parallel tracking and mapping for small AR workspaces. In *Proceedings of the Sixth IEEE and ACM International Symposium on Mixed and Aug-*

mented Reality (ISMAR'07), Nara, Japan, November 2007.

[Kozlov *et al.*, 2007] Alex Kozlov, Bruce MacDonald, and Burkhard Wünsche. Towards improving slam algorithm development using augmented reality. In *Proceedings of the Australian Conference on Robotics and Automation*, 2007.

[Lienhart and Maydt, 2002] R. Lienhart and J. Maydt. An extended set of haar-like features for rapid object detection. In *Proceedings of the 2002 International Conference on Image Processing.*, volume 1, pages 900–903, 2002.

[Lourakis and Argyros, 2004] M.I.A. Lourakis and A.A. Argyros. Vision-based camera motion recovery for augmented reality. In *CGI '04: Proceedings of the Computer Graphics International (CGI'04)*, pages 569–576, Washington, DC, USA, 2004. IEEE Computer Society.

[Marchand and Spindler, 2005] E. Marchand and F. Spindler. Visp for visual servoing: a generic software platform with a wide class of robot control skills. *IEEE Robotics & Automation Magazine*, 12(4):40–52, 2005.

[Mikolajczyk *et al.*, 2005] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. Van Gool. A comparison of affine region detectors. *International Journal of Computer Vision*, 65(1-2):43–72, 2005.

[OGRE, 2008] OGRE. Ogre 3d : Object-oriented graphics rendering engine., 2008. http://www.ogre3d.org.

[Özuysal *et al.*, 2007] Mustafa Özuysal, Pascal Fua, and Vincent Lepetit. Fast keypoint recognition in ten lines of code. In *2007 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2007)*, Minneapolis, Minnesota, USA, 18–23 June 2007.

[Player/Stage, 2008] Player/Stage. The player/stage project., January 2008. http://playerstage.sf.net/.

[Shi and Tomasi, 1994] Jianbo Shi and C. Tomasi. Good features to track. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 593–600, 1994.

[Simon and Berger, 2002] G. Simon and M.-O. Berger. Pose estimation for planar structures. *IEEE Computer Graphics and Applications*, 22(6):46–53, 2002.

[Simon *et al.*, 2000] G. Simon, A.W. Fitzgibbon, and A. Zisserman. Markerless tracking using planar structures in the scene. In *Proceedings of the IEEE and ACM International Symposium on Augmented Reality, 2000. (ISAR 2000).*, pages 120–128, 2000.

[Skrypnyk and Lowe, 2004] I. Skrypnyk and D.G. Lowe. Scene modelling, recognition and tracking with invariant image features. In *Third IEEE and ACM International Symposium on Mixed and Augmented Reality, 2004. ISMAR 2004.*, pages 110–119, 2004.

[Stilman *et al.*, 2005] Michael Stilman, Philipp Michel, Joel Chestnutt, Koichi Nishiwaki, Satoshi Kagami, and James Kuffner. Augmented reality for robot development and experimentation. Technical Report CMU-RI-TR-05-55, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, November 2005.

[Sugimoto *et al.*, 2005] Maki Sugimoto, Georges Kagotani, Hideaki Nii, Naoji Shiroma, Masahiko Inami, and Fumitoshi Matsuno. Time follower's vision: a teleoperation interface with past images. *IEEE Computer Graphics and Applications*, 25(1):54–63, January–February 2005.

[Tomasi and Kanade, 1991] Carlo Tomasi and Takeo Kanade. Detection and tracking of point features. Technical report, Carnegie Mellon University, 1991.

[Wagner and Schmalstieg, 2007] Daniel Wagner and Dieter Schmalstieg. Artoolkitplus for pose tracking on mobile devices. In *Computer Vision Winter Workshop 2007*, February 6-8 2007.

[Young *et al.*, 2006] J.E. Young, E. Sharlin, and J.E. Boyd. Implementing bubblegrams: The use of haar-like features for human-robot interaction. In E. Sharlin, editor, *IEEE International Conference on Automation Science and Engineering, 2006. CASE '06.*, pages 298–303, 2006.