EG UK Theory and Practice of Computer Graphics (2007)
Ik Soo Lim, David Duce (Editors)

A Flexible Approach to High Performance Visualization Enabled Augmented Reality

C. J. Hughes[1] and N. W. John[1]

[1]University of Wales, Bangor

Abstract

Commonly registration and tracking within Augmented Reality (AR) applications have been built around computer vision techniques that use limited bold markers, which allow for their orientation to be estimated in real-time. All attempts to implement AR without specific markers have increased the computational requirements and some information about the environment is still needed. In this paper we describe a method that not only provides a flexible platform for supporting AR but also seamlessly deploys High Performance Computing (HPC) resources to deal with the additional computational load, as part of the distributed High Performance Visualization (HPV) pipeline used to render the virtual artifacts. Repeatable feature points are extracted from known views of a real object and then we match the best stored view to the users viewpoint using the matched feature points to estimate the objects pose. We also show how our AR framework can then be used in the real world by presenting a marker-less AR interface for Transcranial Magnetic Stimulation (TMS).

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Augmented Reality (AR), High Performance Visualization (HPV), Grid

1. Introduction

Augmented Reality (AR) is a field of computer research in which computer-generated artifacts are placed into the user's view of the real world. These artifacts must be correctly orientated with the viewing direction of the user who typically wears a suitable Head Mounted Display (HMD). AR is a technology growing in popularity in medicine, manufacturing, architectural visualization, remote human collaboration, and the military [Azu97, ABB*01].

To create the illusion of a virtual artefact within the real world, it is essential that the virtual object is accurately aligned and that the computer graphics are rendered and presented in real time. Most of the existing solutions involve the use of bold markers that contain contrasting blocks of colour and shapes making them easily identifiable using computer vision techniques. To align virtual artifacts into the real world three main stages are required - see figure 1. Firstly we need to examine the user's viewpoint and identify where our virtual objects belong in the scene. Secondly we need to track the object to ensure that we have aligned the object to the correct position. Finally we use pose estimation to calculate the orientation of the object so that we can align it with the real world.

The Human Interface Technology Laboratory at the University of Washington has developed the ARToolkit, a software library providing the tools for creating marker based AR applications. The ARToolkit has provided the foundation for many of the early developments in AR and make it possible to rapidly produce AR applications using an inexpensive webcam and an average specification PC [KB99].

Moehring, Lessig and Bimber show that when using markers that have high contrast (for example, they use a bold type on a white background), little processing power is actually required to estimate the pose of an object even with the poor capture quality and low processing capability of a standard consumer mobile phone [MLB04].

Although the use of markers in optical tracking enables the pose estimation to be calculated relatively easily, having to use specific markers can limit the possible AR

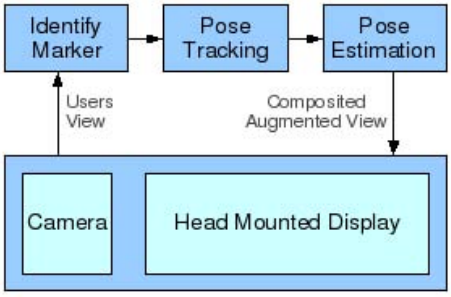

Figure 1: *The three stages involved in AR.*

tools that can be made available. Therefore there are now many examples of AR solutions that do not require markers [CMC03, GRSN02, SK01].

In order to be successful, marker-less tracking is not only more computationally intensive, but also requires more information about the environment and the structure of any planes or real objects that are to be tracked. In this paper we present a flexible solution that does not rely on the use of markers, but rather feature points that are intelligently extracted from the users view. We also provide a solution to the computationally intensive task of pose estimation and the rendering of complex artifacts by exploiting remote HPV resources through an advanced environment for enabling visual supercomputing - the e-Viz Project [RWB*05a, BBC*07].

2. Background

In order to align our virtual object with the real world, we first need to define the object in the users view. During a calibration stage the user is given the opportunity to specify where the object exists within the viewpoint. We use a robust feature point detection algorithm to identify the points that can be repeatedly identified within the space occupied by the virtual object and use this information to estimate the objects position and orientation. There are many existing methods for extracting feature points, most of which are based on corner detection algorithms. Since the 1970's many feature point detectors have been proposed and there is still work today to improve their accuracy and efficiency. There are three main methods for detecting feature points in images, which stem from the following methods: edge-detection, topology and autocorrelation [ZFKP00, Hod98]. It is generally accepted that the autocorrelation methods yield the most repeatable results and they all follow the following three steps:

- For each point in the input image a Cornerness value is calculated by the operator, and relates to how likely it is believed that that point is a corner.
- A threshold value is used to disregard any points that are identified but are not strong enough to be true corners. The Cornerness value of these points is then typically set to zero.
- Non-maximal suppression sets the Cornerness value for each point to zero if its cornerness value is not larger than the cornerness measure of all points within a certain distance. This ensures that we only find maximum points and so we can then assume that all non-zero points are corners.

2.1. Moravec/ Harris algorithms

A very basic algorithm for feature point detection was proposed by Moravec in 1977 as part of his work on machine vision enabled robotics [Mor77, Mor79]. He proposed that a point could be identified as a feature point if there was a significant intensity variation in each direction from that point. Although this algorithm provides basic feature detection without being too computationally intensive, the points it finds are only repeatable when the edges are at 45° or 90° to the point being evaluated. The Harris algorithm [HS88] improves the Moravec algorithm but at a significant cost to the computational requirements. It becomes more robust by changing the way intensity variation is calculated between each pixel and its neighbours by allowing for edges that are not at 45° or 90° to the point being evaluated.

The Harris algorithm uses first order derivatives to measure the local autocorrelation of each point. A threshold value is then used to set all of the weaker points to zero leaving all of the non zero points to be interpreted as feature points - see figures 2 and 3.

More complex methods exist for robust feature point detection, such as the SIFT algorithm [Low04] which could provide further improvements in detecting feature points in non synthetic images, however we have found that the Harris algorithm has provided enough accuracy without the increasing computational load.

2.2. Pose Tracking and Estimation

In order to track the object within our user's view a Cascaded Haar Classifier is used to match the the object to that of a source training image. Our application extends upon previous work [WF06, VJ01] by using multiple Haar classifiers in parallel to detect and track different views of the object.

The Open Source Computer Vision Library (OpenCV) provided by Intel(r) [ope07] provides a basic framework for implementing Haar Classifiers. By using it to implement each of our Haar Classifiers as seperate processes we are able to exploit parallelism for tracking each of the different views of our virtual object in real-time. Each classifier is trained independently using a map of feature points extracted from a source image, each showing different views of the same object. The operator must then set a calibration for each view by manually aligning the virtual artifact with

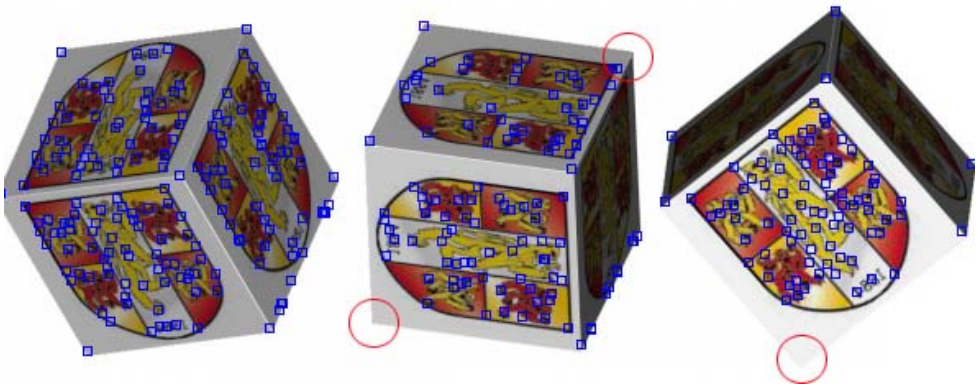

Figure 2: *The Moravec Algorithm. A rotating cube with clearly visible corners is used to show that the feature points are not always repeatable when the cube has moved. The circles clearly indicate points which should have been repeated from the initial pose.*

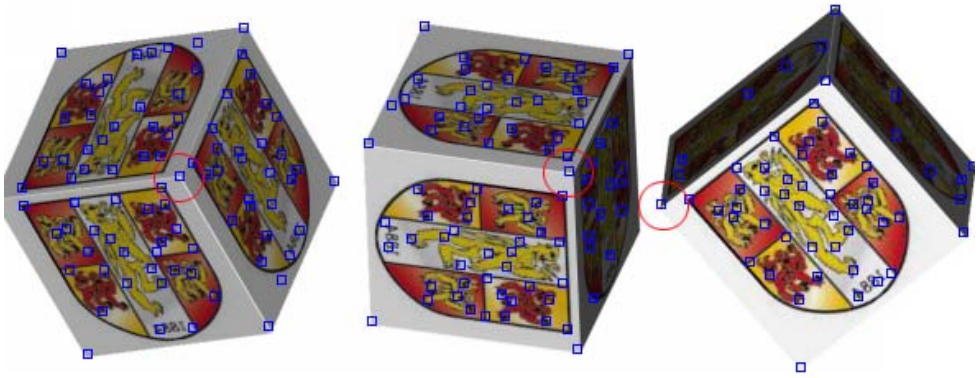

Figure 3: *The Harris algorithm detecting repeatable feature points. The circles show a single point which has been accurately repeated in each movement of the cube.*

the real object in the source image. This calibration information is then stored as a transformation matrix and used later to offset our virtual object to match the arbitrary angle of the camera.

During tracking, the users viewpoint is captured by a camera and a real time feature point map is created and used to query each of the Haar Classifiers. Each classifier then attempts to find a match for the feature points. As soon as each classifier has completed its search the results are correlated and the classifier with the most likely viewpoint of the object is identified. Each matching feature point is identified along with its position in both the source training image and the current view.

A POSIT [DD95] algorithm is used to approximate the perspective projection between the two sets of feature points and therefore find the rotation matrix and the translation vector of the movement between the previously identified source training image and the current camera viewpoint. Our calibration matrix is then applied to this transformation to give a final approximation for the pose of the object and this information is used to align the virtual object with the real world.

Real-time performance is maintained, even with the increased computational load by distributing the pose estimation as part of our visualization pipeline.

3. Utilizing HPV with e-Viz

We are collaborating in the e-Viz project [RWB*05a, BBC*07], which is currently under development and aims to provide a flexible flexible infrastructure for remote visualization using the Grid [FK99]. e-Viz address many of the issues involved in HPV [RWB*05b] using a combination of intelligent scheduling for new rendering pipelines and the monitoring and optimisation of running pipelines, all based on information held in a knowledge base. This provides an

adaptive visualization service that provides rendered graphics reliably without the application or user even being aware of what resources are being used. It also allows the application to render graphics in real time at a resolution that would normally be too high for the client machine.

We have followed two paths for implementing our application with e-Viz:

3.1. Rendering the graphics with e-Viz

The first implementation simply uses the e-Viz framework to render the virtual artifacts present in our AR view. The user's viewpoint is captured by the local machine and the pose estimation is calculated locally. The pose transformation is used to steer the e-Viz visualization pipeline, which in the background sends the transformation information to an available visualization server. Our client then receives the rendered image and composites it locally into the users view - see figure 4.

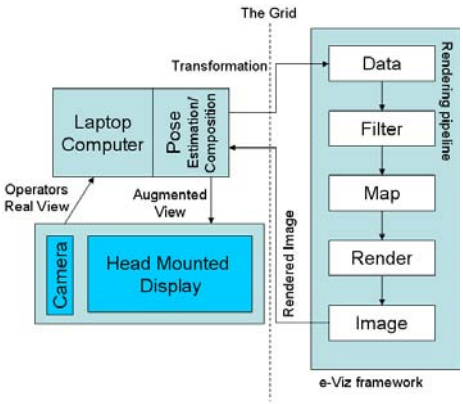

Figure 4: *Using the e-Viz framework to render the graphics but carrying out the pose estimation locally.*

3.2. Distributing the pose estimation module as part of the visualization pipeline.

In order to fully take advantage of the resources managed by the e-Viz framework the second implementation of our software moves the pose estimation module onto the e-Viz visualization pipeline. In this case the e-Viz visualization is steered directly by the video stream of the users view. e-Viz distributes the pose estimation module to a suitable and available resource. The pose estimation module then steers the visualization pipeline and returns the final view back to the user after compositing the artificial rendering into the real scene - see figure 5.

3.3. The e-Viz API

e-Viz provides a client application that can be used to control a remote visualization pipeline as well as providing a viewer for the remotely rendered graphics to be returned to the user. It also provides an API that allows users to develop their own client applications which can utilize the e-Viz managed resources.

The e-Viz framework uses a web service to decide which hardware and software to make available to the client, based upon what resources are needed and what resources are available. The broker uses a knowledge base to store the status of the available servers and inventory what resources they are capable of providing. The client can interact with the Broker by the use of gSOAP calls, which will tell the Client which visualization servers to connect to.

There are generally multiple visualization servers within the e-Viz environment. Having discovered which visualization servers to use, the Client application uses a Grid middleware (such as GT2 /citefoster99globus) to connect to the remote server and run the visualization task. By providing a wrapper to different visualization applications it makes it possible to execute your visualization pipeline on any visualization server regardless of what visualization software it is running.

4. Exemplar application

Transcranial Magnetic Stimulation (TMS) is the process in which electrical activity in the brain is influenced by a pulsed magnetic field. Common practice is to align an electromagnetic coil, on the subject's head, with points of interest identified at a particular location on the surface of the brain. The coil then generates a magnetic field that stimulates this area of the brain, helping researchers identify further information about the brain's function. TMS has also proved to be very useful in therapy and had positive results with treating severe depression and other drug resistant mental illnesses such as epilepsy [Ett98, VM99].

The main challenge for the clinician is to accurately place the coil on the subject's head so that the correct area of the brain is stimulated. In previous work we developed an AR interface for TMS using an optical tracking system to calculate the pose of the subjects head relative to user's viewpoint [HJ06]. We are now developing a new AR interface that uses the e-Viz framework - see figure 6. By removing the need for expensive optical tracking equipment our software will provide an a more flexible solution, making the procedure more accessible to training and further research.

Our research has shown that whilst an average desktop PC does struggle with the pose estimation, using remote resources can ensure real-time performance. Provided the visualization server is appropriate for the rendering task the e-Viz framework is able to return the rendered artefact to the user at a reliable 15 FPS, where there is little latency. On congested networks e-Viz uses stricter compression algorithms at a cost to the image quality to try and maintain these usable frame rates.

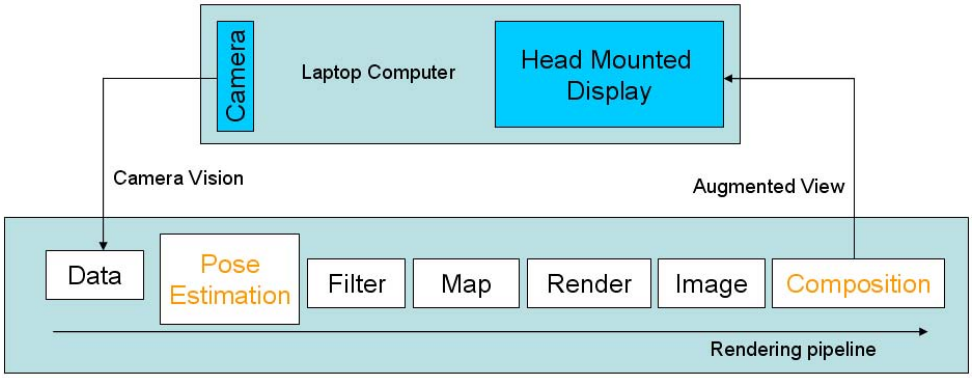

Figure 5: *Implementing the pose estimation module as part of the visualization pipeline.*

5. Conclusions

In conclusion we have found that our approach to producing a framework for AR has been very successful, provided that optimum conditions are available - once the training process is complete there are currently no methods for updating the source images causing any changes in environment to degrade the accuracy of the pose estimation.

Problems occur when trying to run the pose estimation locally. On a typical desktop PC (AMD Athlon 3200, 1Gb RAM) it is simply too computationally intensive and so our example application with four trained views is unable to achieve more than 1-2 FPS. Distributing this calculation to a more powerful parallel grid resource has solved this problem.

Coherence is maintained between frames provided that an accurate calibration is performed. We do however experience a significant jump in the image when switching between the viewpoints used to train the Haar classifier. This can be minimized by allowing the software to only change viewpoint when it needs to rather than always looking for the best match, which can cause it to change erratically when the operators view is bordering between two trained viewpoints. Further methods for averaging between the two found views could be used to smooth over these transitions.

Future work will concentrate on improving the efficiency and reliability of the feature point detection algorithms, ensuring that we have more accurate pose estimation between frames. We also need to introduce heuristics that will help predict the position of the virtual artefact, even if we are unable to calculate the pose of the object, by building up a knowledge base from previous frames. [HJ06]

We have demonstrated a particular application of our AR solution to TMS. However, a wide variety of other applications areas can also benefit. One further example where we currently plan to use this approach is with the Bangor Mixed Reality Anatomy Teaching Tool [TJL06].

6. Acknowledgements

This research is supported by the EPSRC within the scope of the project: "An Advanced Environment for Enabling Visual Supercomputing" (GR/S46567/01). Many thanks to the e-Viz team at Manchester, Leeds and Swansea for their help and support.

We would like to thank Jason Lauder and Bob Rafal from the School of Psychology, University of Wales, Bangor, for allowing us access to their TMS laboratory and for supplying data from past experiments.

References

[ABB*01] AZUMA R. T., BAILLOT Y., BEHRINGER R., FEINER S., JULIER S., MACINTYRE B.: Recent advances in augmented reality. *IEEE Computer Graphics and Applications, 21, 6, 34-47* (2001).

[Azu97] AZUMA R. T.: A survey of augmented reality, presence: Teleoperators and virtual environments. *Vol 6, No. 4 (August), pp. 355-385* (1997).

[BBC*07] BRODLIE K., BROOKE J., CHEN M., CHISNALL D., HUGHES C., JOHN N., JONES M., RIDING M., ROARD N., TURNER M., WOOD J.: Adaptive infrastructure for visual computing. *In Proc. Theory and Practice of Computer Graphics 2007 (To Appear)* (2007).

[CMC03] COMPORT A., MARCHAND Ã., CHAUMETTE F.: A real-time tracker for marker-less augmented reality. *IEEE and ACM International Symposium on Mixed and Augmented Reality, pp. 36-45* (2003).

[DD95] DEMENTHON D. F., DAVIS L. S.: Model-based object pose in 25 lines of code. *Int. J. Comput. Vision 15, 1-2, pp. 123-141.* (1995).

[Ett98] ETTINGER G.: Experimentation with a transcranial magnetic stimulation system for functional brain mapping. *Medical Image Analysis, 2(2):133 - 142* (1998).

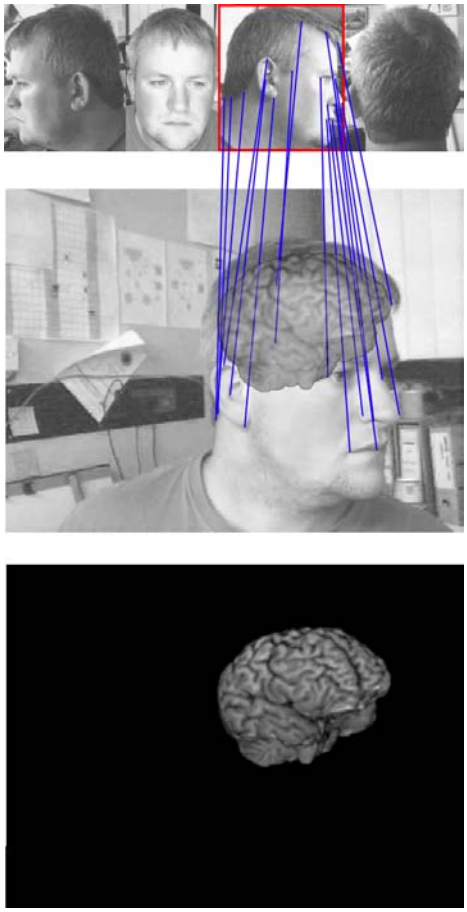

Figure 6: *(a) The four images used to train the Haar classifier with the best match highlighted. (b) The real-time users view with lines illustrating some of the matched feature points. (c) The e-Viz remote rendering view.*

[FK99] FOSTER I., KESSELMAN C. E.: The grid: Blueprint for a new computing infrastructure. *Morgan Kaufmann* (1999).

[GRSN02] GENC Y., RIEDEL S., SOUVANNACONG F., NAVAB N.: Marker-less tracking for ar: A learning-based approach. *IEEE and ACM International Symposium on Mixed and Augmented Reality, pp. 295-304* (2002).

[HJ06] HUGHES C. J., JOHN N.: A flexible infrastructure for delivering augmented reality enabled transcranial magnetic stimulation. *Medicine Meets Virtual Reality, Long Beach, California, pp219-114* (2006).

[Hod98] HODGES K. I.: Feature-point detection using distance transforms: Application to tracking tropical convective complexes. *American Meteorological Society* (1998).

[HS88] HARRIS C., STEPHENS M.: A combined corner and edge detector. *Alvey Vision Conf., Univ. Manchester, pp. 147-151, 1988* (1988).

[KB99] KATO H., BILLINGHURST M.: Marker tracking and hmd calibration for a video-based augmented reality conference system. *In 2nd IEEE and ACM International Workshop on Augmented Reality, San Francisco USA, pp.85-94* (1999).

[Low04] LOWE D.: Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision, 60, 2, pp. 91-110* (2004).

[MLB04] MOHRING M., LESSIG C., BIMBER O.: Video see-through ar on consumer cell-phones. *Ismar, pp. 252-253, Third IEEE and ACM International Symposium on Mixed and Augmented Reality* (2004).

[Mor77] MORAVEC H. P.: Towards automatic visual obstacle avoidance. *5th International Joint Conference on Artificial Intelligence, pp. 584* (1977).

[Mor79] MORAVEC H. P.: Visual mapping by a robot rover. *International Joint Conference on Artificial Intelligence, pp. 598-600* (1979).

[ope07] Intel(r) open computer vision library. *http://www.intel.com/research/mrl/research/opencv/* (2007).

[RWB*05a] RIDING M., WOOD J., BRODLIE K., BROOKE J., CHEN M., CHISNALL D., HUGHES C., JOHN N., JONES M., ROARD N.: e-viz: Towards an integrated framework for high performance visualization. *UK e-Science All Hands Meeting, pp. 1026-1032* (2005).

[RWB*05b] RIDING M., WOOD J., BRODLIE K., BROOKE J., CHEN M., CHISNALL D., HUGHES C., JOHN N., JONES M., ROARD N.: Visual supercomputing - technologies, applications and challenges. *Computer Graphics Forum, Vol. 24 Issue 2, pp. 217-245* (2005).

[SK01] STRIKER D., KETTENBACH T.: Real-time and marker-less vision-based tracking for outdoor augmented reality applications. *IEEE and ACM International Symposium on Augmented Reality, pp. 189-190* (2001).

[TJL06] THOMAS R. G., JOHN N. W., LIM I. S.: A mixed reality anatomy teaching tool. *Theory and Practise of Computer Graphics, vol.24, pp. 117-126* (2006).

[VJ01] VIOLA P., JONES. M.: Rapid object detection using a boosted cascade of simple features. *Computer Vision and Pattern Recognition, pp. 511-518* (2001).

[VM99] V. W., M. R.: A primer of magnetic stimulation as a tool for neurophysiology. *Neuropsychologia, Peergamon, Feb;37(2):125-35* (1999).

[WF06] WILSON P. I., FERNANDEZ J.: Facial feature detection using haar classifiers. *J. Comput. Small Coll. 21, 127-133* (2006).

[ZFKP00] ZITOVA B., FLUSSER J., KAUTSKY J., PETERS G.: Feature point detection in mulitframe images. *Czech Pattern Recognition Workshop* (2000).