

## Designing and Experimenting with a Distributed Tracking System

Girish G. Joshi, Rajeev R. Raje, Mihran Tuceryan  
*Department of Computer and Information Science*  
*Indiana University Purdue University Indianapolis*  
*Indianapolis, USA 46202-5132*  
 {ggjoshi, rraje, tuceryan}@cs.iupui.edu

### Abstract

*Tracking objects is an important activity in many applications such as Augmented Reality, Visual Servoing, and Tangible Interfaces. Most of these applications are inherently dynamic and demand real-time response while tracking. Also, due to the issues of economics, such applications can benefit from tracking systems that do not depend on well-engineered setups and instead use inexpensive sensors for tracking. These requirements make distributed tracking a challenging problem that needs a thorough investigation. In this paper an approach to tracking, which uses the concepts of dynamic discovery, in a distributed environment containing autonomous vision-based trackers that use inexpensive sensors (such as Web-cameras) is presented along with its empirical validation. The empirical results obtained from experiments are promising and validate the premise that Distributed Tracking Systems can be built using inexpensive vision-based sensors and concepts of dynamic resource discovery.*

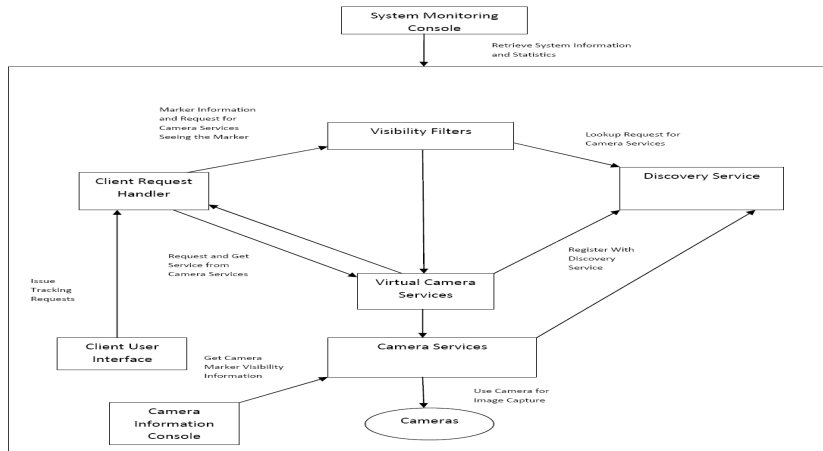
### 1. Introduction

Tracking an object using a vision-based sensor device such as a camera means continuously identifying its location and orientation when either the object or the camera is moving [1]. Many applications such as Augmented Reality [2, 3], Visual Servoing [4], and Tangible Interfaces [5] are being created using tracking as an important component. If tracking is to be used for these applications, it has to be carried out with a real-time response and preferably in setups which are not well-engineered for tracking (setups where users do not have much control over the environment and are not allowed to modify the environment) [6]. Also, tracking applications demand that the tracking environment should allow arrivals and departures of sensors as well as the objects being tracked. Prevalent approaches to tracking include Active Badge [7],

InterSense Tracking technologies [19], Active Bat [8], HiBall [17], ART trackers [18], Cricket [9], RADAR [10], Easy Living [12], Smart Floor [13], and MotionStar DC magnetic tracker [11]. Most of these approaches assume a well-engineered environment, where the environment in which tracking is performed has to be controlled or altered prior to achieving tracking. Also, many of them require expensive equipments to be in place to carry out the position tracking and need a centralized controller which has to be present in the setup to perform the tracking activity. Adding new trackers and achieving a real-time response time for tracking is a difficult issue that is recognized by many of these systems. This paper describes a Distributed Tracking System (DTS) that aims to address these limitations and challenges. The DTS uses inexpensive vision-based sensors (such as Web cameras) and the concepts of spontaneous resource discovery so as to tackle the challenges of real-time response, unknown setups, and the dynamic nature of the tracking environment. A prototype of the DTS is created and is empirically validated to assess its capabilities. Results obtained from an extensive experimentation are discussed in the paper. The rest of the paper is organized as follows: the architecture of the DTS is presented in the next section; it is followed by implementation details along with the experimental setup in section 3; section 4 presents the categories of experiments carried out in the DTS; and the paper concludes with the lessons learned and future work to be carried out.

### 2. Architecture of the DTS

Figure 1 depicts the overall architecture of the DTS. Cameras used in the DTS are widely available and are inexpensive Web cameras. The Camera Services encapsulate these physical Web cameras and provide a layer of abstraction on top of the physical Cameras.



**Figure 1. Design of the DTS and associated activity flow**

The Camera Services are responsible for returning the pose of the object being tracked in the form of a homogeneous Transformation Matrix, along with a time-stamp. The DTS assumes that the object being tracked has a specific Marker associated with it. The transformation matrix is a 4 X 4 matrix which gives the translations and rotations required in order to transform from the Camera Coordinate system to the Marker Coordinate system. Given the camera position and orientation as a part of the reply to a query for transformation matrix, the position and orientation of the Marker can be computed using the transformation matrix by a simple matrix multiplication. In order to meet the real-time constraints, a concept of Marker registration is used with the Camera Services, which enables a Camera Service to trigger up the background evaluation of the Marker Visibility even before any request to find the transformation matrix is issued.

The Virtual Cameras are services that are built on top of the Camera Services. The purpose of creating Virtual Cameras is to allow for assessing the scalability of the DTS during the empirical validation. In order to assess the scalable nature of the DTS, it is necessary to have hundreds of Camera services in the experimental setup, an option that is not feasible for the moment. An easier solution, as proposed in the DTS, is to employ a few Web Cameras and associated Camera Services and build many Virtual Camera Services on top of these Camera Services. The Camera Services, Virtual Camera Services, and the Visibility Filters (described shortly) are registered with the Discovery Service, which provides the spontaneous discovery of these entities based on a tracking query.

A Client (the object to be tracked), which enters into the DTS, discovers the Discovery Service through unicast messages and then using the Discovery Service, discovers the appropriate Visibility Filter(s) and the Virtual Camera Services. The Visibility Filter uses the discovery service to find a set of Virtual Camera Services available in the DTS. Discovery Service enables entering and leaving of Services and Clients in and out of the DTS and thus, reduces need for a well-engineered setup.

In this setup, it is assumed that markers are attached rigidly to the objects to be tracked. Thus, the ability to track markers means the objects of interest can be tracked by applying a simple rigid transformation to the marker pose. Camera Services perform the task of tracking these Markers associated with the objects to be tracked. Once a Camera captures an image, it has to locate the Marker within the image and then do the estimation of the pose, i.e., the transformation from the Camera to the Marker coordinate system. The Markers used in the DTS are black squares with distinctive two-dimensional shapes used as identifiers similar to 2D barcodes [15]. The information that the Virtual Camera and Camera Services need about the Marker is the bitmap image, i.e., the representation of the Marker image in numeric values. The Marker information can be presented to the Virtual Camera and Camera Services, which can then use that information to locate a Marker within the Captured Image. Thus, the Visibility of the Marker is decided by the Virtual Camera and Camera Services.

The Visibility Filter is built on top of the Discovery Service and helps the Clients to locate the Virtual Camera Services which are able to track the marker at the given instant of time. Upon a Client request to find a set of Virtual Camera Services seeing a given Marker(s), the Visibility Filter has to identify this set

out of all Virtual Camera Services that are present at that instant in the DTS. The Visibility Filter continuously keeps on locating the Virtual Camera Services and their Visibility Information. Upon a client query, it just uses the local information to do the computation. This speeds up the response time of the lookup activity similar to the technique used by the Camera Services. Visibility Filter also provides the concept of registration of Markers so as to start evaluation of Camera-Marker Visibility information prior to the first request by the client. The Visibility Filter uses the registration facility provided by the Camera Services to register the Markers with the individual Camera Services which enables the Camera Services to start evaluating Marker visibility information. To avoid the Visibility Filter becoming a bottleneck in the system, the DTS has multiple Visibility Filters spread across the tracking setup any of which can be used by the Clients to get the set of Virtual Camera Services seeing the Markers.

Clients are being tracked in the DTS. As indicated earlier, Clients employ markers to get the tracking information by using the Discovery Service, the Virtual Camera Service and the Visibility Filter. Clients can be a variety of devices such as, PDAs, Laptops, or even Desktop Computers. The DTS provides the Client User Interface through which the User can interact with the system and get the tracking information. The Client User Interface issues requests from the Users to the Client Request Handler which takes care of the User requests to track the Marker(s). The Client Request Handler first discovers the Discovery Service in the DTS using unicast messages and then discovers a set of Visibility Filters available in the DTS and decides which Visibility Filters are to be used to get the set of Virtual Cameras seeing the Markers. Once the Client Request Handler gets the set of Virtual Cameras seeing the Markers, it queries a subset of them to get the transformation matrices of the Cameras relative to the Markers, the position of the Cameras, computes its own position with respect to all the Cameras and fuses the positions using a simple averaging fusion to get a more accurate position. The Client Request Handler also uses the Client User Interface to show the position information to the User.

The Camera Information Console provides information about the Web Cameras and their current Marker visibility in the DTS on providing the Camera Service name and the machine on which it is running. This Console provides a view of the DTS from the perspective of the Cameras. The System Monitoring Console provides information and statistics about the various entities in the system namely, the Camera Services, Virtual Camera Services and the Visibility Filters. It shows the names and locations (host names)

of the Camera Services. It also displays the names and locations of Virtual Cameras and the Camera Services that they are using. It shows the names and locations of Visibility Filters in the system. In this way, the System Monitoring Console provides the overall statistics of the system.

### **3. Implementation and Experimental Setup**

The ARToolkit [15] provides the API to detect a given Marker in a captured camera image. A Java-binding for ARToolkit is available called JARToolkit [16]. It is used in implementing the above DTS. The JINI [14] lookup service provides the necessary discovery service mechanism required to allow spontaneous networking, i.e., arrival and departure of Services and Clients seamlessly to and from the DTS. All the entities of the DTS are written in Java language using Java Platform, Standard Edition 6 Development Kit (JDK 6) and the discovery service is written using the JINI Technology starter kit is 2.1. The basic setup for all these experiments consists of 3 physical Cameras hosted on three machines running on Windows XP (Version 2002, Service Pack 2) connected by a 10 Mbps Local Area Network, 3 Camera Services accessing these 3 cameras on 3 machines, 3 Visibility Filters one per machine and varying number of Virtual Camera Services, Varying Number of Clients and Varying Number of Markers.

### **4. Experimentation**

The metrics measured in the experimentation with the DTS are three Response Times, namely, Visibility Filter Response Time (abbreviated as VFRT), Camera Service Response Time (abbreviated as CSRT) and End-To-End Response Time (abbreviated as EERT). The VFRT is the time taken by the Visibility Filter to respond to a Client query for Virtual Camera Services which can see the given set of Marker(s). The CSRT is time taken by the Camera Service to return the transformation matrix for a given set of Marker(s). The EERT is the addition of times taken to search for Virtual Camera Services that are able to see the Marker(s), to get the tracking information from each of those selected Virtual Camera Services and then to fuse the information using a simple averaging fusion. The real-time nature of the DTS demands that all these response times should be small enough – the rate of image capture and processing should be at least 30 frames/second.

#### **4.1. Service Specification, Types of Queries**

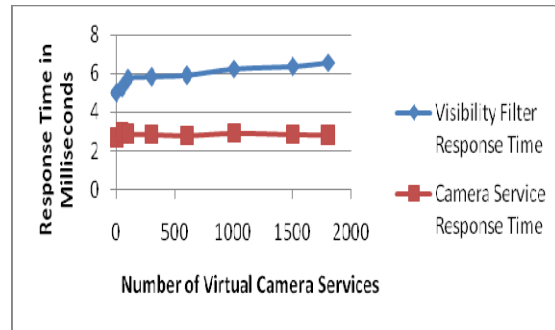
All the services in the DTS are specified using string names (such as “CameraM1” or “M1\_VirtualCam\_1” or “UFilterM1”), which are provided by the owner of the service and have a type associated with them. The types are “CamService”, “VirtualCamera” and “UFilter” for Camera Service, Virtual Camera Service, and Visibility Filter respectively. The Queries used in the experimentation are of two types: query sent from the Client or the Visibility Filter to the Virtual Camera Service to get the transformation matrix for the given set of Marker(s) and query sent from the Client to the Visibility Filter to discover a set of Virtual Camera Services which are seeing the given set of Marker(s). Both these query types consists of name(s) of the Marker and the URL from which the Marker information can be downloaded as a part of the query.

#### 4.2. Performance and Scalability Experiments

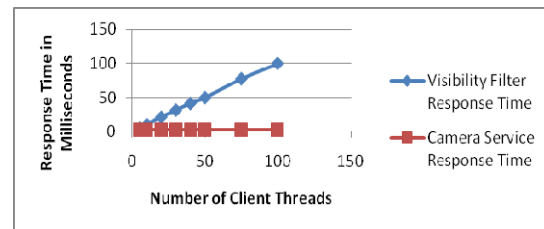
The experiments described in this section test the effect of changing the number of Virtual Camera Services, the number of Clients threads, the number of Markers in the DTS, the number of Virtual Camera Services returned by the Visibility Filter on the VFRT and CSRT. The experiments also test the effect of changing the number of Virtual Camera Services used for fusion, and the number of Markers used in the DTS on the EERT. The experiments involve queries sent in the following manner: the Client has 5 threads each of which sends 100 queries each. In the experiments analyzing the effect of increase in the number of Markers, the Client threads divide the given set of Markers among themselves (in a round robin fashion). The other experiments which do not involve multiple Markers have every Client Thread using the same Marker for the Query. The measurements from 4 iterations of the experiment and 100 queries per experiment are averaged to arrive at the final average value for a particular point on the line graph. Results of the experiments and associated analyses are presented below.

Figure 2 shows that as the number of Virtual Camera Services in the DTS increases, the VFRT increases by a small amount and the CSRT is unaffected by the change in the number of Virtual Camera Services. The reason behind the increase in VFRT is that the Visibility Filter has to update the Marker Visibility Information about more number of Virtual Camera Services. As a result, the time required to process the request from a Client increases. The CSRT is relatively unaffected because a presence of additional Virtual Camera Services does not

significantly alter the processing time of the existing Virtual Camera Services.



**Figure 2. Effect of number of virtual camera services on VFRT and CSRT**

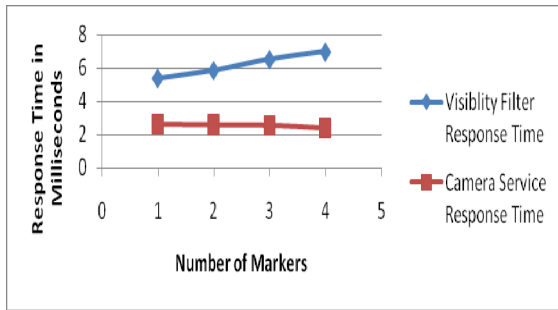


**Figure 3. Effect of number of client threads on VFRT and CSRT**

Figure 3 shows that there is a steady increase in VFRT with the increase in the number of Client threads whereas the CSRT remains unaffected. The reason for the increase in the VFRT is that as the number of Client threads increase, the Visibility Filter is subject to more load with the increase in Client threads (and thus, the input requests). Also, there is a contention between the threads to get the CPU and as a result, the VFRT increases with increase in the number of Client Threads. The rationale for CSRT not to get much affected is that there is not enough computation involved in the Virtual Camera Service or the Camera Service for retrieving the Camera-Marker visibility information.

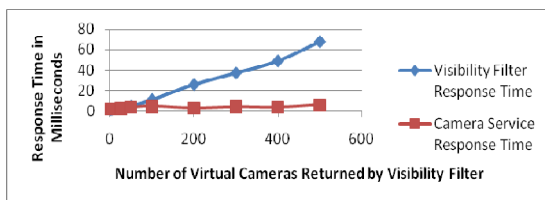
Figure 4 shows that the VFRT increases uniformly with the increase in number of markers being tracked and again, the CSRT remains unaffected. The reason for the increase in VFRT is that as the number of Markers that are being detected in the DTS increases, the background thread updating the values of Marker visibility for each Virtual Camera Service in the Visibility Filter has to perform more computation, thereby, increasing the VFRT. The CSRT does not get affected much by the increase in the number of Markers because as explained earlier, there is not a great amount of computation involved in a Camera

Service getting the transformation matrix and even if the number of Markers increases, the increase in the computation in the background thread is not substantial, i.e., the background thread will have to compute the visibility for multiple Markers instead of one and this does not take a considerably larger amount of time so as to affect the CSRT.



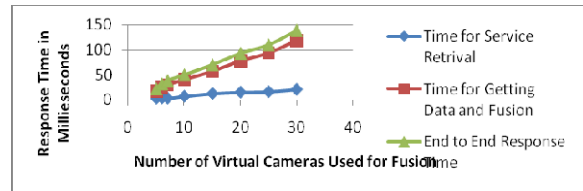
**Figure 4. Effect of number of markers on VFRT and CSRT**

Figure 5 shows that increasing the number of Virtual Camera Services returned by the Visibility Filter clearly increases the VFRT but it does not have any effect on the CSRT. The reason for the increase in VFRT is that in order to return a larger number of services the Visibility Filter needs to perform more look up and matching operations. But the CSRT has no such problem, as it depends upon calls to each individual camera service and hence, it does not get affected by the number of services returned by the Visibility Filter.



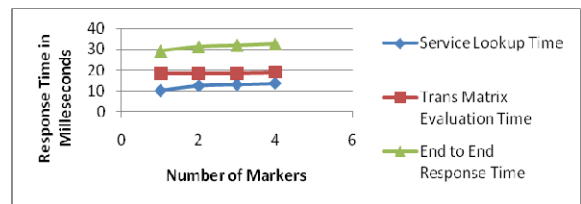
**Figure 5. Effect of number of virtual camera services returned by the visibility filter on VFRT and CSRT**

Figure 6 shows the variations of EERT and its components, namely, the time for retrieving Virtual Camera Services and the time for Getting the Tracking Data and Fusion against the number of Virtual Camera Services to be used for fusion.



**Figure 6. Effect of virtual camera services used for fusion on EERT**

As the number of services to be fused increases, the EERT increases because as more number of services are to be returned back to the Client, the Client needs to get the tracking information from these services and then perform a fusion operation on these different values to get the final tracking information. From Figure 6, it can be seen that if a real-time response (i.e., 33 ms/ frame) is desired, values from only up-to 6 services can be fused. Thus, for the current prototype of DTS, the upper limit for the fusion is 6 Virtual Camera Services.

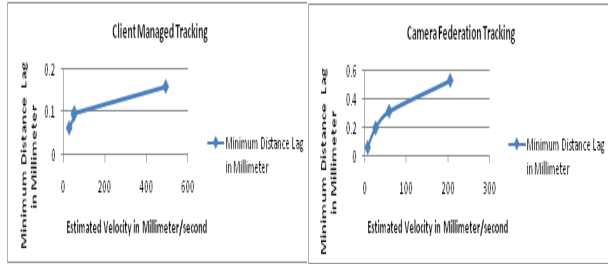


**Figure 7. Effect of number of markers on the EERT and its components**

Figure 7 shows that an increase in the number of Markers used in the DTS increases the EERT slightly. The evaluation time for the transformation matrix, which is the number of Virtual Camera Services to be fused multiplied by the time to evaluate one transformation matrix, remains more or less steady but the time to look up the Virtual Camera Services from Visibility Filter increases slightly, the combined effect results in slight increase in the response time EERT with increase in the number of Markers.

### 4.3. Experiments testing the effect of motion of the Marker on the accuracy of the DTS

This category of experiments observes the effect of moving Markers on the accuracy of DTS. In a tracking environment where the Markers move, the services offered by the Visibility Filters are used more frequently as the set of Virtual Camera Services which can spot the Marker keeps changing continuously as the Marker moves.

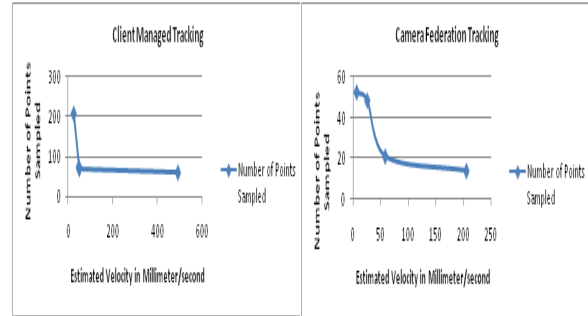


**Figure 8. Effect of velocity of marker motion on minimum distance lag**

The accuracy with which the DTS can provide the tracking information can be tested much better with Moving Markers as compared to stationary ones because there can be multiple points at which a Marker will be spotted by the Cameras and accuracy can be measured across various such readings. These experiments have two sub-categories. In the first sub-category of experiments, the clients have the responsibility for finding different Virtual Camera Services from the Visibility Filter as the Marker moves along and then use those Virtual Camera Services to get tracking values. The second sub-category of experiments involves the formation of a Federation of Virtual Camera Services (Virtual Camera Services talking to one another) to perform tracking of the Marker(s). During the tracking activity, a Virtual Camera Service that finds other Virtual Camera Services which can see the Marker(s) obtains the tracking information from these other Virtual Camera Services and then fuses them to get more accurate tracking information. Once the Marker goes out of the sight of the original Virtual Camera Service, it tries to look for some other Virtual Camera Services which can see the Marker currently. If it finds such Virtual Camera Services, it will pass the responsibility of tracking that marker to this newly found Virtual Camera Service and stops its tracking of that marker. The accuracy of the tracking depends upon the number of points or the tracking information that the clients are able to sample for the moving Marker. The accuracy can be also defined by a parameter which this paper calls the Minimum Distance Lag. The Minimum Distance Lag is the minimum distance between any two successive readings or points where the Marker was spotted by the Camera Services. The smaller the value of minimum distance lag, the better is the accuracy of tracking. The velocity of the moving Marker can also be estimated by the total distance travelled and total time taken to cover that distance.

Each of these categories of experiments involves moving the Marker in front of the cameras at four different speeds and measuring the velocity, number of sampled points and Minimum Distance Lag. Figures 8

and 9 show the results of these experiments. Figure 8 shows the variation of Minimum Distance Lag and Figure 9 shows variation of Number of Points Sampled against both the estimated velocity of the moving Marker. From the Figures 8 and 9, it can be observed that as the estimated velocity of the moving Marker increases, the number of points sampled decreases whereas the Minimum Distance Lag increases.



**Figure 9. Effect of velocity of marker motion on number of points sampled**

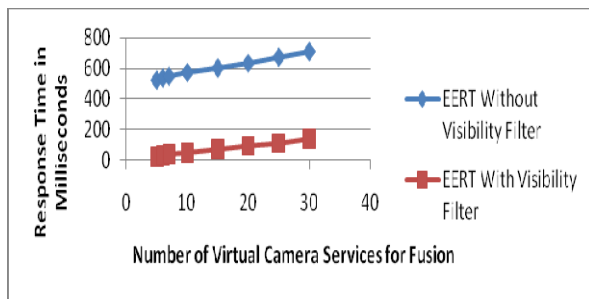
The rationale behind this is that as the velocity of the motion of the Marker increases, the Camera Service has less time to capture its image because every time it goes to capture the next image, the Marker has already moved ahead greater distance with increases in velocity. One more trend which is seen from these graphs is that the Client managed tracking samples more points and has a comparatively less value of Minimum Distance Lag as compared to the Federation of Camera experiment. This is because, in the Federation of Camera setup, once the Marker moves out of the sight of the one Camera into the sight of the other Camera, the responsibility of tracking is shifted from one Camera to the other and this leads to missing some of the sampled points which the Client managed tracking would have managed to sample because there is only one Client instead of a set of Cameras tracking the Marker. This fact also leads to a comparatively more value of Minimum Distance Lag in case of Federation of Camera experiment as compared to Client Managed Tracking.

#### 4.4. Experiments testing the Fault Tolerance of the DTS

These experiments present the semantics of the DTS in case of failure and the effect on the EERT due to the failures of the Visibility Filter, Virtual Camera Services, Camera Services, the Physical Camera and the Client.

In the case when the Visibility Filter has failed, the Client has the burden to discover the Virtual Camera

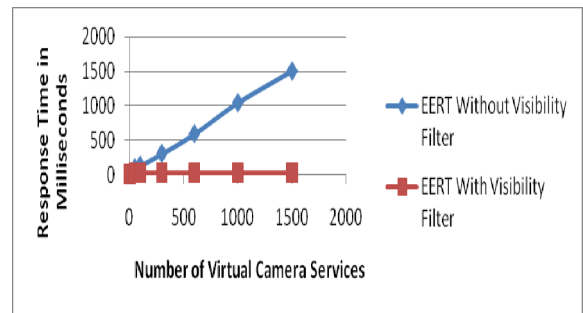
Services, get Marker Visibility information from them, and then select a set of Virtual Camera Services to perform the tracking activity. These experiments are carried out to show the variation of the EERT in absence of the Visibility Filter with two parameters; the number of Virtual Camera Services used for fusion to estimate the Client position and the number of Virtual Camera Services present in the DTS. As with the experiments testing performance and scalability of the DTS, each point on the results graph is arrived at after averaging out values from 4 iterations (with a the same set of parameters) of each experiment which involves the Client with 5 client threads pumping 100 queries each. The first experiment observes the effect of increasing the number of Virtual Camera Services used for fusion on the EERT in absence of Visibility Filter and also compares it with earlier the experiment which tested the same effect in presence of the Visibility Filter. The second experiment observes the effect of increasing the number of Virtual Camera Services in the DTS on the EERT in absence of the Visibility Filter and compares it with the results obtained in the presence of the Visibility Filter. As with the earlier experiment, a similar trend of an increase in the EERT is observed with the increase in the number of Virtual Camera Services used for fusion – this behavior is shown in Figure 10.



**Figure 10. Effect of number of virtual camera services used for fusion in presence and absence of visibility filter on EERT**

The reason for the increase in the EERT in absence of the Visibility Filter is that the Client has to fuse information from increasing number of Virtual Camera Services and this leads to increase in the EERT. As seen from Figure 10, the value of the EERT is considerably large in absence of the Visibility Filter as compared to the corresponding value of the EERT in the presence of the Visibility Filter due to the additional computation, of discovering, selecting and fusing, that the Client needs to perform to achieve the functionality that was offered by the Visibility Filter. The overall effect leads to significant increase in the EERT for the Client in the absence of Visibility Filter.

From results of the second experiment shown in Figure 11, it can be seen that the EERT increases linearly with an increase in the number of Virtual Camera Services in the DTS in absence of the Visibility Filter. The reason behind this behavior is that as the number of Virtual Camera Services increases, the Client discovers and has to fetch more number of Virtual Camera Services which increases the EERT. Again, the value of the EERT is quite large in absence of the Visibility Filter as opposed to the corresponding value in the presence of the Visibility Filter. This is again attributed to the presence of background thread in the Visibility Filter which the Client does not have. There is a slight increase in the EERT (although not clearly seen in the Figure 11 due to the scale used for plotting) with an increase in the number of Virtual Camera Services in the presence of the Visibility Filter. This is due to an increase in the number of Virtual Camera Services present in the DTS, as observed in earlier experiments on scalability and hence, the EERT also increases because EERT involves the time (VFRT) to fetch Virtual Camera Services from the Visibility Filter.



**Figure 11. Effect of number of virtual camera services in presence and absence of visibility filter on the EERT**

## 5. Conclusions and Future Work

This paper proposes an approach for design and implementation of a Distributed Tracking System using vision-based inexpensive sensors such as Web cameras and concepts of spontaneous resource discovery. A series of experiments testing the Scalability, effect of moving Markers, and Fault tolerance has been performed on the DTS. It can be observed from the results of these experiments that the DTS is scalable with respect to the Number of Virtual Camera Services, the Number of Clients, the Number of Markers used as the VFRT and CSRT do not increase a lot due to increase in any of these parameters and allows a graceful degradation of the performance due to the incorporation of the failure

semantics. All the experiments help to validate the premise of the thesis that DTS can be created using vision-based inexpensive sensors such as Web-cameras, concepts of spontaneous service discovery and concepts of pre-fetching and background evaluation of Camera Visibility information. Several future extensions to this research work are possible. For example, the service specifications can be more extensive incorporating the Quality of Service features and thereby, allowing the possibility of more rigorous selection of Camera Service while tracking objects. Also, a generic framework can be built on top of this version of DTS for the Sensors to talk to each other and perform tracking whenever some Sensor spots an incoming object of interest. In addition, methods such as Kalman Filtering can be used to carry out fusion of tracking information provided by different sensors to get more accurate tracking information.

## 6. References

- [1] Lepetit, V., and Fua, P., “Monocular Model-Based 3D Tracking of Rigid Objects: A Survey”, *Foundations and Trends in Computer Graphs and Vision*, Vol. 1, No. 1, 2005:1-89.
- [2] Azuma, R., Baillot, Y., Behringer, R., Feiner, S., Julier, S., and MacIntyre, B., “Recent advances in augmented reality”, *IEEE Computer Graphics & Applications*, Vol. 21, No. 6, 2001:34–47.
- [3] Klinker, G., Ahlers, K., Breen, D., Chevalier, P., Crampton, C., Greer, D., Koller, D., Kramer, A., Rose, E., Tuceryan, M., and Whitaker, R., “Confluence of computer vision and interactive graphics for augmented reality”, *Presence: Teleoperators and Virtual Environments*, Vol. 6, No. 4, 1997:433–451.
- [4] Hutchinson, S., Hager, G. and Corke, P., “A tutorial on visual servo control”, *IEEE Transactions on Robotics and Automation* Vol. 12, No. 5, 1996:651–670.
- [5] Ullmer, B., and Ishii, H., MIT Media Lab, Tangible Media Group, “The metaDESK: Models and Prototypes for Tangible User Interfaces”, In the Proceedings of UIST '97, October 14-17, 223-232, 1997, © 1997 ACM
- [6] Azuma, R., Lee, J., Jiang, B., Park, J., You, S., and Neumann, U., “Tracking in unprepared environments for augmented reality systems”, *Computers & Graphics* Vol. 23, No. 6, Dec 1999:787-793.
- [7] Want, R., Hopper, A., Falcão, V., and Gibbons, J., “The Active Badge Location System,” *ACM Transactions Information Systems*, 91-102, Jan 1992.
- [8] Harter, A., Hopper, A., Steggles, P., Ward, A., and Webster, P., “The Anatomy of a Context-Aware Application,” *Proceedings of 5th Annual. International Conference on Mobile Computing and Networking (Mobicom 99)*, ACM Press, New York, 59-68, 1999.
- [9] Priyantha N., Chakraborty A., and Balakrishnan H., “The Cricket Location-Support System,” *Proceedings of 6th Annual. International Conference on Mobile Computing and Networking (Mobicom 00)*, ACM Press, New York, 32-43, 2000.
- [10] Bahl, P., and Padmanabhan V., “RADAR: An In-Building RF-Based User Location and Tracking System,” *Proceedings of IEEE Infocom 2000*, IEEE CS Press, Los Alamitos, Calif., 775-784, 2000.
- [11] Technical Description of DC Magnetic Trackers, Ascension Technology Corporation, Burlington, Vt., 2001.
- [12] Krumm, J., Harris, S., Meyers, B., Brumitt, B., Hale, M., and Shafer, S., “Multi-Camera Multi-Person Tracking for Easy Living,” *Proceedings of 3rd IEEE International Workshop on Visual Surveillance*, IEEE Press, Piscataway, N.J., 3-10, 2000.
- [13] Orr, R., and Abowd, G., “The Smart Floor: A Mechanism for Natural User Identification and Tracking,” *Proc. 2000 Conf. Human Factors in Computing Systems (CHI 2000)*, ACM Press, New York, 275 – 276, 2000.
- [14] Sun Microsystem Inc. Jini technology architectural overview. Technical report, Sun Microsystem, Inc, 1999, site: <http://www.sun.com/software/jini/whitepapers/architecture.pdf>.
- [15] ARToolKit, 2002, site: <http://www.hitl.washington.edu/artoolkit>
- [16] Geiger, C., Reimann, C., Sticklein, J., and Paelke, V. , “JARToolKit - A Java binding for ARToolKit”, In the Proceedings of Augmented Reality Toolkit, The First IEEE International Workshop, 5-5, 2002.
- [17] Welch, G., Bishop, G., Vicci, L., Brumback, S., Keller, K., and Colucci, D.: “The HiBall tracker: High-performance wide-area tracking for virtual and augmented environments”. In: *Proceedings of the ACM Symposium on Virtual Reality Software and Technology*, London, 1-ff, 1999.
- [18] Advanced Realtime Tracking GmbH, 1999, site: <http://www.ar-tracking.de/>
- [19] InterSense Inc, 1996, site: <http://www.isense.com/>