Fast Color Fiducial Detection and Dynamic Workspace Extension in Video See-through Self-Tracking Augmented Reality

Youngkwan Cho Jun Park Ulrich Neumann
Computer Science Department
Integrated Media Systems Center
University of Southern California
Los Angeles, CA 90089, USA
{ ykcho | peterpan | uneumann }@graphics.usc.edu
http://tracker.usc.edu/~ykcho/star.html

Abstract

The registration problem is one of the major issues in Augmented Reality (AR). Fiducial tracking is gaining interest as a solution to this problem in video see-through AR because of the availability of digitized real scenes. There are several AR systems using fiducial tracking, but most of them operate in small desktop workspaces. It is difficult to apply them directly to large scale applications. The wide range of work distance and non-uniform lighting conditions make fiducial detection very difficult. Adding new fiducials requires off-line process for measuring positions of new fiducials. In this paper, we propose a fast and robust fiducial detection procedure with carefully designed color fiducials and noise analysis of digitized images. We also present a dynamic workspace extension method with on-line position determination of unknown features. Our efforts is a framework for applying AR to large scale applications.

1. Introduction

In Augmented Reality (AR), virtual objects have their own volumes and locations in the user's real world. To add virtual objects on the user's view, AR uses two types of cameras: a real camera and a virtual camera. A real camera generates an image stream of the user's real world and a virtual camera adds 3D graphics of virtual objects on the live image stream. Misalignment of these cameras makes virtual objects appear incorrectly on the composite images. Registration between virtual objects and the real scene is one of the major issues in AR [Azuma94] [Bajura95][Tuceryan95].

To solve the registration problem, both of the cameras should maintain the same camera pose (position and orientation) as well as internal parameters. Several types of tracking devices have been used to get the pose of a real camera. Error in a tracking device is inevitable and it is the major source of misregistration. In video see-through AR, a user sees the real world through a video camera, and the same video stream is also available to the AR system in digital format. There have been several efforts to detect and correct misregistration by tracking fiducials in the digital video stream [Azuma95][Bajura95][State96] (Fig. 1 (a)). Moreover, a video see-through AR system can extract the camera pose directly from the video stream when it finds enough fiducials in each image [Mellor95] [Kutulakos96][Neumann96] (Fig. 1 (b)).

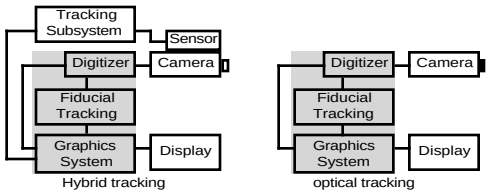


Figure 1. Video See-through AR system with optical fiducial tracking

Figure 1. Video see-through AR system with fiducial tracking

Fiducial tracking should be processed robustly in real time to achieve a high refresh rate with stable system operation, but it is a very computationally expensive process. Most of the developed systems operated in small desktop workspaces with a small number of fiducials (in most cases less than ten) [Bajura95][Mellor95] [State96] [Kutulakos96]. The sizes of fiducials in the images were large, and the lighting conditions were pretty uniform, making fiducial detection and identification relatively easy and fast. But it is difficult to apply these constraints directly to large scale applications with tens or hundreds of fiducials and non-uniform lighting conditions. These conditions put the robust fiducial detection and identification problem into a very different dimension.

Edge detection and segmentation of gray-scale image have been active areas in computer vision. There have also

been several works done with color images [Kender76] [Nevatia77][Celenk91][Healey92]. These works were interested in whole images, and they usually could not work in real-time. Because our interest is extraction of a few fiducials in real-time, we need a different approach which does not require the whole image analysis. We propose a fast and robust fiducial detection procedure with carefully designed fiducials and noise analysis of digitized images in varying light conditions.

The fiducial set is pre-designed according to the application environments, and fiducials are carefully prepared before the system operation. When a user finds a work area without enough fiducials, he could stop the operation and add more fiducials in the area of interest. In this paper, we present a dynamic workspace extension method with on-line 3D position determination of unknown features. This capability also helps the robust camera pose calculation by adding some features which are consistently detected through multiple frames taken from different view points.

In this paper, we present a framework for applying AR to large scale applications. In section 2, we discuss a fast and robust fiducial detection procedure with a widely applicable fiducial set and noise analysis of digitized images. In section 3, we provide a reliable on-line method to determine 3D positions of unknown features for dynamic workspace extension. Our results and discussion are provided in section 4.

2. Fiducial detection

Fiducial detection is the first step of fiducial tracking and a very computationally expensive process. A high refresh rate (more than 10 frames per second) is desired for comfortable video see-through AR. All processes from the image digitization to the composite image generation should be completed within the interval between frames. For stable system operations, fiducial detection should find all proper fiducials in the input image obtained from a suitable camera pose.

2.1. Characteristics of fiducials

Any detectable features can serve as fiducials, but carefully designed fiducials greatly help fast and robust detection and identification. Desirable properties of good fiducials include

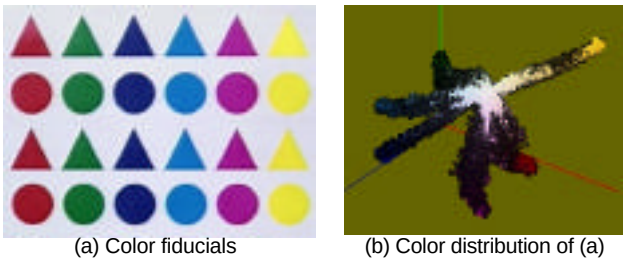
1) fast and robust detection and identification
2) small not to block any important information in the real world
3) inexpensive and easy application.

Active fiducials (e.g., LED) are easily detectable because they are usually brighter than environments, and their intensities are constant [Ward92][Bajura95]. Since they are 3D objects and require power, it is difficult to install them and to determine their centroids exactly.

Mellor used concentric black and white circular fiducials [Mellor95]. Corners of large black rectangles were used in [Kutulakos96][Koller97]. Fiducial detection only deals with intensity, and it is fast. But fiducial identification is difficult because all fiducials are identical. Big rectangles occupy large portion of an image.

Multi-colored concentric circle sets are helpful for robust fiducial detection and identification [State96]. Because the colors of outer ring and center circle are mixed on the boundary, it is hard to identify them correctly from a distance. They usually require much larger size than solid colored fiducials for robust detection and occupy large portion of an image.

We tried various multi-colored and shaped fiducials. It was hard to calculate the centroids and the sizes of fiducials with multiple colors and complex shapes. They also required large size for robust detection. We concluded that the most widely applicable fiducials were solid colored 2D circles and regular triangles in six colors (red, green, blue, yellow, cyan, and magenta) [Neumann96] (Fig. 2(a)). Solid colored fiducials are more robustly detectable from a distance than multi-colored fiducials of the same size. The circle has been widely used because of its symmetry, view invariance, and easy verification. The triangle is a simple and easily verifiable shape, and is not easily confused with the circle. The six suggested colors are far apart in the RGB color space, and they are not easily confused with each other in the presence of noise. The suggested fiducial set has 12 unique color and shape combinations, which make fiducial identification much easier. This fiducial set satisfies most of the desirable properties of good fiducials.



(a) Color fiducials (b) Color distribution of (a)

Figure 2. Color fiducials and color distribution in the color cube (see color plate)

2.2. Digitized image analysis

A fiducial detection procedure extracts fiducials from the digitized image taken with a color video camera. Since the camera and digitizer are not immune to noise, intensity and color changes across the boundary of a fiducial should be

discriminated from those due to noise. Fixed global thresholds would not work for non-uniform lighting conditions. It is useful to analyze the noise characteristics of digitized images for dynamic local threshold determination.

2.2.1. Color distance metric. Each pixel has three color components (red, green, and blue) in a digitized color image. Intensity-only analysis is not enough for detection of the suggested color fiducials, and all three color components should be used. Since we expect intensity and color changes across the boundary of a fiducial, we need metrics to measure intensity and color differences between two pixels. Intensity $\boldsymbol{Y}$ at a pixel is easily calculated with a linear combination of three color components.

$$Y = aR + bG + cB,$$

where $0 \le a \le 1$, $0 \le b \le 1$, $0 \le c \le 1$, and $a + b + c = 1$. R, G, and B are red, green, and blue value at the pixel, respectively. a, b, and c are assigned depending on the selected color coordinate system. We used equal weights (a=b=c=0.333). The intensity difference between two pixels is trivial; subtract the small intensity value from the big one.

Because color information consists of three values, a color distance metric requires an effective method to combine six values. There are several possible metrics.

1) Absolute distance, such as Manhattan distance, Euclid distance, etc.
2) L^1 normalized distance [Kender76][Wyszecki82]
3) L^2 normalized distance [Healey92]
4) Angle between colors in the RGB color cube [Sung92]

Absolute measures are not appropriate because adding low amplitude noise in a bright area of the RGB color cube does not affect color perception but adding the same noise may give different color perception in a dark area. L^1 normalized distance depends on color difference as well as the colors themselves. L^2 normalized distance and color angle are good metrics, but they require computationally expensive square root and trigonometric operations. Therefore they are not suitable for real-time applications.

We employed the square of the color angle cosine. It is monotonically decreasing for the range 0° - 90° and fast to compute.

$$\cos^2(\theta) = \frac{(color_1 \cdot color_2)^2}{\|color_1\|^2 \|color_2\|^2}$$

Like other normalized metrics, this metric has a problem with the color black whose R, G, and B values are 0, 0, and 0, respectively. Around black objects and very dark shadows in an image, the intensity can be only a measure.

2.2.2. Camera and digitization model. Charge-Coupled Device (CCD) cameras are popular as imaging devices in many areas. Healey and Kondepudy proposed a CCD camera model and noise estimation [Healey94]. The quantization process for a single collection site was modeled as

$$D = (KI + N_{DC} + N_S + N_R) A + N_Q,$$

where KI is the number of collected electrons, N_{DC} is the number of electrons due to dark current, N_S is the Poisson shot noise, N_R is the amplifier read noise, A is the combined gain of the output amplifier and the camera circuitry, and N_Q is a zero mean quantization noise.

We simplified the model $D = \mu + N$, where μ is the mean and N is a zero mean noise. To analyze the mean behavior and noise characteristics, we fixed the camera toward the color fiducials on a sheet (Fig. 2(a)), and took ten pictures at each intensity setting under daylight, fluorescent light, and combined light conditions. The analysis procedure of the pictures and the results are explained in the following subsections.

2.2.3. Color behavior analysis. The fiducial pixels in the pictures taken under the combined lights were divided into groups according to their colors and intensities. The averages of groups were computed. The same process was repeated with the pictures taken under daylight and fluorescent light. Three results gave very similar patterns but the actual values were a little different. Three graphs in Fig. 3 show the average behaviors of the RGB channels of our camera as functions of pixel intensity under the combined light. Each curve in the graphs shows the channel response of the corresponding color. For example, to find the response to the red target, follow the red curves in the three graphs. The red curve in the first graph shows the red channel response. The red curve in the second and third graphs show the responses of the green and blue channels to the red target. At low intensities,

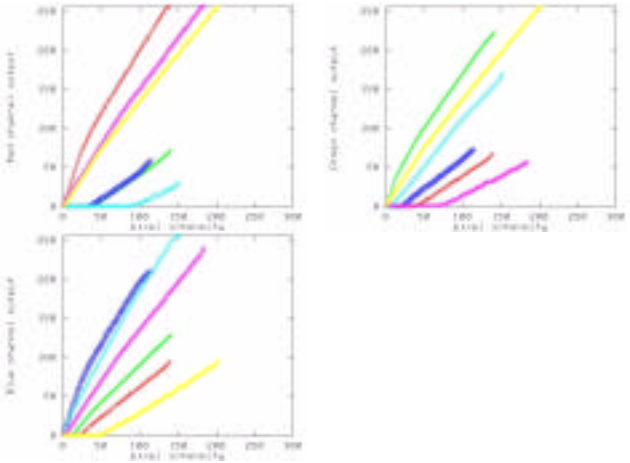


Figure 3. Red, green, and blue color channel behaviors of the six target colors (see color plate)

only the red channel has non-zero values. As the intensity increases, the other channels also have small non-zero values. Therefore a red fiducial at low intensity gives a different direction of color vector from one at high intensity.

These mean color behaviors are captured in the reference colors used for the fiducial color classification in the fiducial detection procedure. As the above result shows, only one set of reference colors at a specific intensity does not work for all intensities. Therefore we need multiple sets of reference colors or sets of color functions. To simplify the representation, we used curve fitting with order-four polynomials.

2.2.4. Intensity noise analysis. We obtained groups of ten pictures taken with the same intensity under the same light source. Because the camera was fixed, each pixel should have the same intensity for pictures in a group if the camera was noise free. Therefore at each pixel, the differences from an average were noise. The differences were collected from all groups according to the average intensity and the color. The statistics of noise were computed for each intensity in each color.

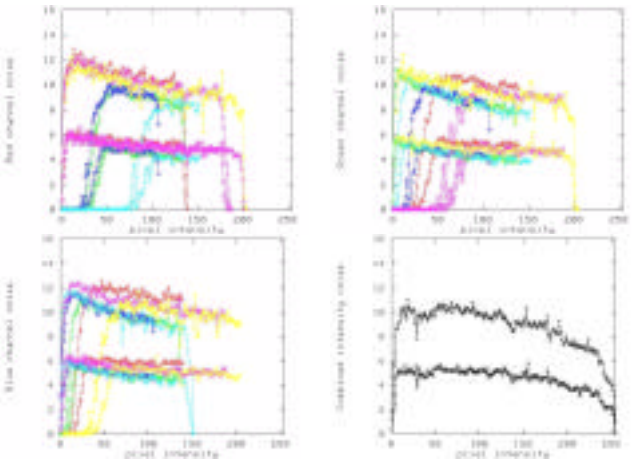
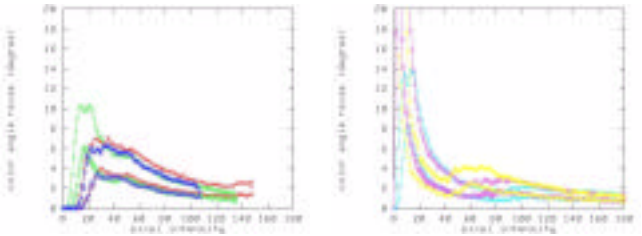


Figure 4. Cumulative 80% and 99% noise distributes for RGB channels and intensity (see color plate)

Fig. 4 shows the cumulative noise distributions in pixel unit of our camera as functions of pixel intensity. The lower curves in each graph are the 80% cumulative noises of corresponding colors, and the upper curves are the 99% cumulative noises. The noises are slightly decreasing, as the intensity of a pixel is increasing. All three color channels in the six target colors show similar patterns and values.

2.2.5. Color noise analysis. Using the same procedure as intensity noise analysis, we calculated the distributions of color angle noise. Fig. 5 shows the plots of cumulative distributions (80% and 99%) in degree units of our camera as functions of pixel intensity. The left graph shows the distributions of primary colors and the right graph shows those of secondary colors. The upper curves are 99% cumulative distributions of corresponding colors, and the lower curves are 80% cumulatives. The color noises are reciprocally related to the pixel intensity. The absolute magnitude of noise at low intensities is slightly larger than at high intensities (Fig. 4), but the relative magnitude is large at low intensity. The color noises of primary colors converge to zero at low intensities because only one channel is non-zero, but those of secondary colors have large values because the multi-channel noises are not correlated.

This color noise analysis shows that a tight threshold would work well in high intensity regions, but thresholds should be loose in low intensity areas. At very low intensities, the color difference should not be used as a discriminating criterion at all.



(a) Primary colors (b) Secondary colors

Figure 5. Cumulative 80% and 99% color noise distributions (see color plate)

These intensity and color angle noise analyses are used for adaptive threshold determination during segmentation. The Figures 4 and 5 suggest that the thresholds should be related to pixel intensity. In addition, we recommend multi-level thresholds to smooth noise. The most sensitive level (e.g., cumulative 80% noise curves) allows only possible noise within a pixel, and it might divide one fiducial into several regions. The second level (e.g., cumulative 99% noise curves) threshold allows the noise from the non-uniform surface, and merges some small regions in a fiducial into large ones. The third level threshold takes account of other factors, such as partially in shadow, partial highlight, camera characteristics (e.g., preamphasis on boundaries), etc.

2.3. Fiducial detection procedure

We are not interested in a whole image but a few fiducials in the image. The fiducial detection process should be done in real-time for a high refresh rate. Therefore we need a different approach from those often used in computer vision. Since there are only a few fiducials in each image, our basic idea is to quickly skip

uninteresting areas and to concentrate only a few potential fiducial regions.

Our detection process is divided into five steps (Fig. 6). We quickly skim the whole image and select potential fiducials in the first step. For those selected regions, we apply more expensive operations in sequence to distinguish real fiducials from false fiducials and to extract information about detected fiducials.

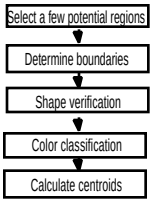


Figure 6 Major steps in fiducial detection

Figure 6. Major steps in fiducial detection

2.3.1. Potential fiducial detection. We expect only a few fiducials in each image because fiducial tracking requires only three or four fiducials. Therefore only a few small regions in the image need to be fully analyzed. When a user does not move or moves smoothly, the potential locations of fiducials can be predicted based on the history of fiducial locations. When the system starts or loses track of the fiducials, it should scan the whole image. It also scans the whole image periodically to check if there are any new features for the new-point problem (section 3).

To find potential fiducials without processing every pixel in the image, we construct a lower resolution image with subsampling and look for potential fiducials in the small image. The subsampled image is small the search is fast but not robust because some appropriate fiducials might be missed. When it is large, the system runs robustly but slow. Therefore the sampling interval is important to achieve fast and robust fiducial detection.

To find all fiducials, we should have at least one sampling point from each fiducial. The sampling points form a square grid in the image. If the grid square is smaller than the maximum square inside a fiducial, at least one sampling point falls on the fiducial. Therefore the sampling interval should not be longer than the side of the maximum inside square of the expected minimum fiducials (Fig. 7). A circular fiducial looks like an ellipse in the image. The length of the major axis d of the expected minimum ellipse is

$$d = DF/L,$$

where D is the diameter of a fiducial, L is the maximum work distance, and F is the effective focal length of a camera. When we allow a user to look at a fiducial $\Theta°$ from its surface normal at the maximum work distance, the length of the minor axis d' of an ellipse is

$$d' = d\cos(\Theta) = DF\cos(\Theta)/L.$$

The side length of the maximum inside square of the ellipse is

$$l = \sqrt{\frac{d^2 d'^2}{d^2 + d'^2}} = \frac{DF}{L}\sqrt{\frac{\cos^2(\Theta)}{1+\cos^2(\Theta)}}$$

When the minimum fiducial size is available from the previous frame, we adaptively change the sampling interval and improve the performance further.

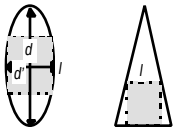


Figure 7 Maximum sampling interval

Figure 7. Maximum sampling interval

The side length of the maximum square inside the expected minimum triangle is computed similar to that of a circular fiducial.

$$l = \frac{D' F}{L}\frac{\sqrt{3}\cos(\Theta)}{\sqrt{3}+2\cos(\Theta)},$$

where D' is the side length of a triangle. To use the same sampling interval with circles, the ratio of the side length of a triangle to the diameter of a circle is

$$\frac{D'}{D} = \frac{\sqrt{3}+2\cos(\Theta)}{\sqrt{3+3\cos^2(\Theta)}}.$$

The subsampled small image is segmented into regions based on the similarity of intensity and color with adaptive multi-level thresholds. After segmentation, we compute the bounding box of each region and select the regions with appropriate size and aspect ratio as potential fiducials (Fig. 8).

Figure 8. Potential fiducial areas

2.3.2. Boundary determination. More intensive tests are applied to each potential fiducial to check if it is a real fiducial. For further analysis, the boundary information of a potential fiducial in full resolution is required. The boundary location on a sampled scanline is easily determined using the boundary information in low resolution. The boundary location on an intermediate scanline is searched from the linearly predicted position with the adjacent known scanlines.

There are several methods for determining the boundary of features with sub-pixel precision in computer vision [Oakley91][Hunt95][Mellor95]. Because of the video

bandwidth limitation, the digitized image does not have step edges, but rather ramps that are several pixels wide. Since our color metric is not linear, we use intensity to determine the sub-pixel boundary location in a ramp. On each scanline the feature intensity and the background intensity are calculated by averaging the stable areas on both sides of the ramp. We find the location in the ramp whose intensity is same as the average of two end intensities.

2.3.3. Shape verification. To discriminate the real fiducials from the false ones, the shapes and colors are used. For shape verification, a series of tests is performed. They are arranged to cull false fiducials as early as possible with simple and fast tests.

Area ratio of feature to bounding box. A circle would appear as an ellipse, and a regular triangle as a general triangle in the image. If the detected feature is a circle and the major or minor axis is vertical, the area ratio of the detected feature to the bounding box is PI/4 = 0.7854. If the axis is not vertical, the ratio is still near to 0.7854. If it is a triangle and one side is vertical or horizontal, the ratio is 0.5. If not, the ratio is different according to a view point, but it is roughly 0.5.

Number of contact points. A circle contacts a bounding box at four points, and a triangle contacts at three points. This test obviously distinguishes a triangle from a circle.

Line test. A rectangle embedded in a circle touches the circle boundary four points.. We check for boundary points of the feature outside of the rectangle. In a triangle, three contact points make a triangle that is identical to the detected feature. We test if boundary of the feature is close to the triangle. For the fast processing, we just test on a few sample points on each side of rectangle or triangle.

2.3.4. Color classification. The color classification of fiducials discriminates false fiducials whose shapes and sizes are similar to those of real fiducials. Some false fiducials, continuously detected through multiple frames, can be used as fiducials after their 3D locations are determined with the new-point determination algorithm (section 3).

RGB equations of the standard colors and color noise approximation are developed in the section 2.2. After calculating average intensity and color of each fiducial, the average color is compared to the standard colors at the average intensity. If the smallest color difference is less than the color noise, the fiducial is assigned to the color (Fig. 9). If not, the fiducial is false one and is passed to the new-point algorithm for 3D location determination.

Figure 9. Detected fiducials

3. 3D Position Estimation of Unknown features

From this point, the term “registered features” will be also used to refer to “fiducials” in contrast with “unknown features” whose 3D positions are not known yet. The camera can be tracked as long as it sees at least 3 fiducials at the same time. So more fiducials provide more viewing freedom in a fiducial tracking system. For example, fiducials should be widely placed to have a large workspace and more densely placed in an area to be viewed closely. For a small number of fiducials in a limited volume, a mechanical digitizer can be used to measure and register their 3D coordinates. Larger volume requires that the digitizer's base needs to be moved, and a coordinate system transformation between the positions must be computed. This process is difficult and erroneous and impossible to perform dynamically.

Fortunately, the camera can be still tracked in a limited range with a small set of registered features (measured using, for example, a digitizer). So the tracking information (camera position and orientation) is available and can be used to compute the 3D positions of unknown features in the neighborhood of registered features. Once the 3D position of a previously unknown feature becomes known and the feature is registered, this feature can also be used to track the camera and again to compute the positions of other unknown features in its neighborhood. Fig. 10 shows the overview of the iterative Position Estimation System.

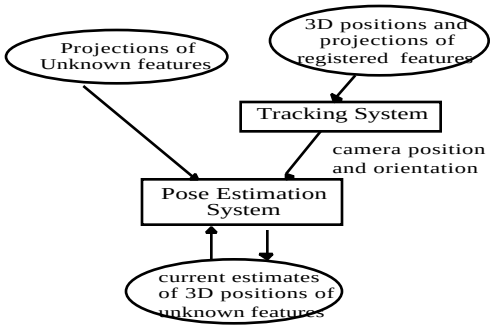


Figure
Pose Estimation System estimates 3D positions of unknown features using camera position/orientation and projections of unknown features on scrren space.

Position estimation system estimates 3D positions of unknown features using camera position/orientation and projections of unknown features on the image plane.

Figure 10. Position estimation system diagram

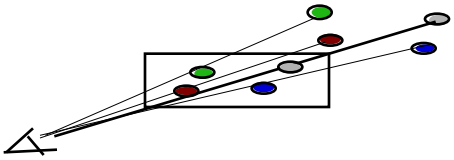


Figure
color feature : known
gray feature : unknown
since camera position and

- black features : registered - used to calculate camera positions and orientations
- gray feature : unknown - the 3D position is at the intersection of two lines connecting camera positions and projections of the feature

Figure 11. Unknown feature position estimation

3.1. System Requirements

Each frame, a line is formed connecting the camera position and the projection of the unknown feature on the image plane. The 3D position of the unknown feature could be anywhere on the line, but two lines from different views, if they intersect, determine a unique 3D position (Fig. 11). However, because of the noise in obtaining the centroids of features and calculating camera positions and orientations, lines do not generally intersect at one point requiring an optimizing algorithm to get a reliable estimation of the 3D position.

In computer vision literature, there has been much work in pose estimation of a camera's viewpoint [Haralick89] and object recognition [DeMenthon95] [Huttenlocher90] where absolute or relative structures of the objects are assumed to be known. However, a system to recover point-wise structures from known but noisy camera pose and image sequences has not been developed as far as we know. There has been a similar previous work in estimating the 3D position of a feature in Robotics [Lee97], but in their case, a stereo camera was used and a 3D position estimate of the feature was available each frame. In our application, a monocular camera is used and information from each frame is insufficient in itself.

The intersection of two lines with a wide angle gives an estimate of a 3D point position which can be used for many applications. But in AR more accurate estimate of a 3D point position will make more accurate camera pose available, and more accurate camera pose will superimpose virtual objects on real objects more correctly. Especially when the workspace needs to be dynamically extended, the propagated error will become bigger even with small error in estimation.

Our goal was to develop a system which converges quickly even if the initial estimate is poor. The computation time should be low in accumulating useful information and updating the estimates. It is also important to keep orientations of lines which can be used to determine how good the accumulated data set is. Even with a same number of frames, the result varies according to the behavior of the camera. Since a user may wear the camera, we can not predict whether the camera moves or rotates evenly to most directions or stays in a narrow angle. The estimates when the camera moves evenly over many directions are better than when the camera movement is restricted in a narrow angle (Fig. 12).

(a) Camera movement is restricted : can not expect to get a good estimate because not enough information has been collected.

(b) Most frames to one direction and small number of frames to other directions : better than (a) but still not enough information has been collected.

(c) Camera moves evenly to most directions: a good data set

Figure 12. Behaviors of the camera

3.2. Algorithm

3.2.1. Estimate Representation. We represent the estimate of the 3D position as a 3D point and its position uncertainty as an ellipsoid. As measurements(lines) contribute to the estimate update, the uncertainty ellipsoid shrinks according to the orientations and proximities of the lines.

3.2.2. Uncertainty Ellipsoid Representation. The uncertainty ellipsoid is represented by a covariance matrix. A covariance matrix has been also used in edge detection to collect orientation information to form edges [Guy95]. The covariance matrix can be decomposed into eigen-vectors and eigen-values which represent the axes and lengths of the uncertainty ellipsoid. Initially, the uncertainty ellipsoid is a sphere whose radius is determined by the standard deviation of the measurement noise, which could be obtained off-line. The uncertainty ellipsoid can be used to inform a directional uncertainty as well as the scalar uncertainty.

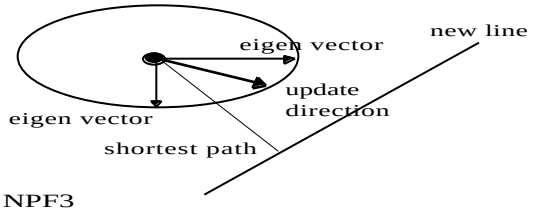
3.2.3. Estimate Update Direction. The 3D point position is estimated based on the collected lines. As a new line is measured, the current estimate needs to be moved toward the new line because the new line infers that the point position is somewhere on the line. Since the previously collected lines are represented by the uncertainty ellipsoid, the update direction and magnitude are determined based on the new line and the uncertainty ellipsoid.

Larger uncertainty value in a direction gives more flexibility and freedom to move toward that direction, the estimate is moved more in a direction with a larger directional uncertainty and less in a direction with a smaller directional uncertainty. One of the ways to reflect this idea is to scale the update direction according to the uncertainty ellipsoid. The update direction is determined as a weighted average of three directions which are suggested by each of 3 axes of the uncertainty ellipsoid.

Let ev_i, i=1,2,3 be eigen-vectors and λ_i, i=1,2,3 be eigen-values of the uncertainty ellipsoid. A direction vector v_i is a vector connecting the current estimate and the closest point(closest from ev_i) on the line for each i. These vectors are the desired movement direction by each axis of the uncertainty ellipsoid. Then the update direction v is average of v_i's weighted by eigen-values(directional uncertainty). See Fig. 13 for a graphical description of the 2D case.

$$v = \sum_{i=1}^{3} \lambda i ? v_i$$

This works well when the input lines are not evenly distributed and uncertainty is biased to one direction which is the typical case in AR.

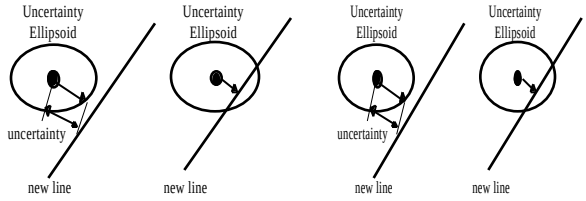


NPF3
the estimate is updated to the direction of shortes path to the line scaled by uncertainty to move more to more uncertain direction

The estimate is updated to a direction toward the line considering the uncertainty ellipsoid.

Figure 13. Direction of estimate update

3.2.4. Estimate Update Magnitude. In updating the estimate, the uncertainty ellipsoid is used to get the uncertainty value in the direction of update. This uncertainty value determines how much the estimate is to be updated. This process reflects the idea of updating more if the estimate is uncertain while updating less if the estimate is certain in that direction. As useful measurements are collected, the uncertainty ellipsoid shrinks and the estimate becomes stable moving less and less. Fig. 14 describes estimate update graphically for the 2D case. The 3D case is similar.



NPF4
Left : the estimate is updated in bound of the uncertainty
Right : the estimate is updated not beyond the line

(a) the estimate is moved in bound of the uncertainty

NPF4
Left : the estimate is updated in bound of the uncertainty
Right : the estimate is updated not beyond the line

(b) the estimate is not moved beyond the line

The estimate is updated by minimum of the uncertainty bound and the distance to the line.

Figure 14. Magnitude of estimate update

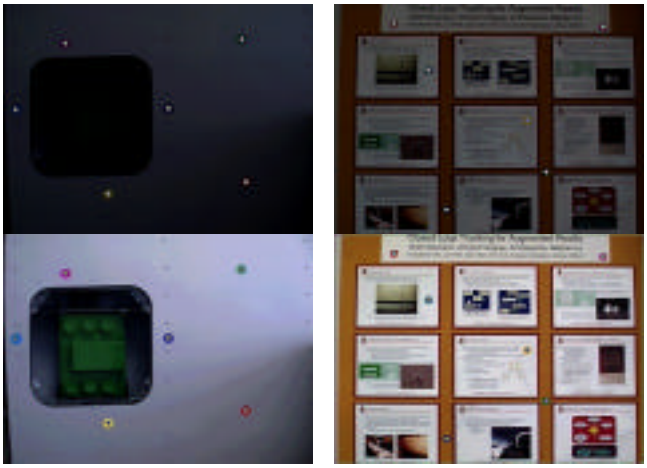
4. Result and Discussion

Our video see-through AR system has the following configuration:

- SGI Indy 24-bit graphics system with MIPS4400@200MHz
- SONY DXC-151A color video camera with 640x480 resolution, 31.4° in horizontal and 24.37° in vertical field of view (FOV), S-video output
- Fiducial set with six colors (red, green, blue, yellow, cyan, and magenta), two shapes (circle and regular triangle), and 1 inch diameter size

We set the work-distance to 2 ~ 8 feet, and allowed a user to look at a fiducial 45° from its surface normal at 8 feet distance. Within this work-volume, A circular fiducial appeared as an ellipse with major axis length 12 ~ 48 pixel in images. Therefore the sampling interval is initially set to 6, and it is adaptively changing according to the minimum fiducial size of the previous frame.

The curves in Fig. 3 were approximated with order four polynomials. We used three levels of intensity and color angle thresholds. Our implementation of the fiducial detection gave 12 frames per second (FPS) when each fiducial is located within a window of the previous location. For whole image search, it processed 7 FPS with simple scene (Fig. 15 (a)) and 5 FPS with complex scene (Fig. 15 (b)). The sampling interval, and the size and number of potential fiducials in the image are the major factors of system performance.



(a) Simple background (b) Complex background

Figure 15. Fiducial detection under various lighting conditions

We tested the position estimation system with a Gaussian distribution of lines about a point and also with real data generated by the camera. Graphical charts in Figures 16 and 17 shows the distances(in inch scale) between the estimates and the real 3D position for each frame. The scalar confidence values are also shown to

inform how much the system is certain of the estimate. The scalar confidence is simply $(1-\sum_{i=1}^{3}\lambda i)$, where $\sum_{i=1}^{3}\lambda i$ is the scalar uncertainty and $\lambda i, i = 1,2,3$ are lengths of ellipsoid axes.

Gaussian Random numbers with standard deviation of 0.2 and 0.4 were used to simulate the system. The result indicates that the estimation(the distance from the estimation to the real position was measured) converges fast and becomes stable as the confidence grows.

Two sets of real data were generated using the camera. A model with 3 registered features and 1 unknown feature on a 3D object was used. The unknown feature was also digitized to compute the distances between the estimates and the digitized value. Data set 1 was selectively collected to get a good data set(it is more evenly oriented), so it converges fast. Data set 2 is a non-selective data where the camera was free-running. In this case the first estimate is not as good as data set 1 because the angle between first two lines was smaller. Estimates of both data sets are compared with averages of intersections of every pair of accumulated lines(all combinations) whose time complexity is $O(n^2)$ (Fig. 17).

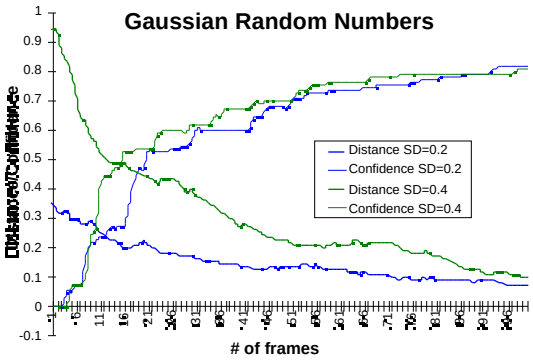
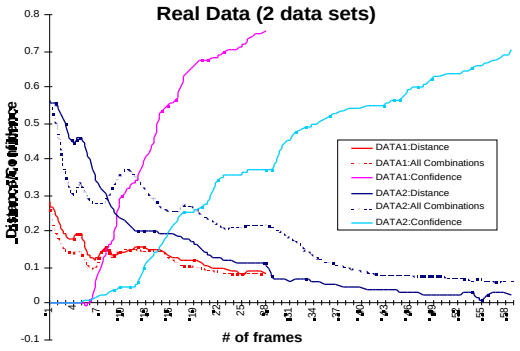


Figure 16. Increasing confidence of point position using synthetic data



distances are in inch scale

Figure 17. Result: real data sets

We presented an algorithm for estimating positions of unknown features for dynamic workspace extension. The system worked well both with synthetic and real data. Our algorithm out-performed the average intersection of every pair of lines in non-selective free-running data set. For the free-running case, it took about 50 frames of contributed lines until it converged and became stable. Considering the accuracy of the mechanical digitizer(0.017"), our system's accuracy is quite good(for data set 1: 0.08" after 30 frames, for data set 2: 0.025" after 50 frames).

This is an extension of the previous system implementation [Neumann96]. After many tests with various fiducials, we believe that the suggested fiducial set is the most generally applicable fiducial set for large scale applications. We can use multi-sized fiducials to prevent all fiducials from become bigger. The new fiducial detection based on the fiducial shape is much faster and more robust than the previous one, which was based on color similarity to standard colors, under various light conditions. The on-line new-point position determination algorithm makes it possible to add new fiducials during the system operations and dynamically extend the workspace.

The presented work in the paper is only a first step for applying AR to large scale applications. There are still many areas that need more development before seriously using AR in real applications. With the proposed fiducial set, the fiducial identification problem is greatly reduced, but it is still a big problem when hundreds of fiducials are used. The three-point camera pose determination gives multiple solutions. When the solutions are collapsed into two, we can choose the right one with fiducial size comparison. But more research is required to deal with the case where the number of solutions is more than two.

References

[Azuma94] Ronald Azuma and Gary Bishop, Improved Static and Dynamic Registration in an Optical See-through HMD, Proceedings of SIGGRAPH 1994, pp. 197-203

[Azuma95] Ronald Azuma, A Survey of Augmented Reality, SIGGRAPH 1995 Course Notes #9

[Bajura95] Michael Bajura and Ulrich Neumann, Dynamic Registration Correction in Augmented Reality Systems, Proceedings of IEEE Virtual Reality Annual International Symposium 1995, pp. 189-196

[Celenk91] M. Celenk, Color image segmentation by clustering, IEE Proceedings-E, Vol. 138, No. 5, ., 1991, pp. 368 - 376

[DeMenthon95] D.F. DeMenthon, and L.S. Davis, Model-Based Object Pose in 25 Lines of Code, IJCV, No. 1-2, June 1995, pp. 123-141

[Fischler81] M.A. Fischler and R.C. Mizell, Random Sample Consensus: A Paradigm for Model Fitting with Application to Image Analysis and Automated Cartography, Communications of ACM, Vol. 24, No. 6, 1981, pp. 381-395

[Guy95] Gideon Guy, Inference of Multiple Curves and Surfaces from Sparse Data, Ph.D Dissertation, University of Southern California, 1995

[Haralick89] R.M. Haralick., H. Joo, C.N. Lee, X. Zhuang, V.G. Vaidya, and M.B. Kim, Pose Estimation from Corresponding Point Data, IEEE Trans. Systems, Man and Cybernetics, No.6, November/December 1989, pp.1426-1446

[Haralick94] R. Haralick, C. Lee, K. Ottenberg, and M. Nolle, Review and Analysis of Solutions of the Three Point Perspective Pose Estimation Problem, International Journal of Computer Vision, Vol. 13, No. 3, 1994, pp. 331-356

[Healey92] Glenn Healey, Segmenting Images Using Normalized Color, IEEE Transactions on Systems, Man, and Cybernetics, Vol. 22, No. 1, Jan./Feb. 1992, pp. 64-73

[Healey94] Glenn E. Healey and Raghava Kondepudy, Radiometric CCD Camera Calibration and Noise Estimation, IEEE trans. on PAMI, Vol. 16, No. 1, Mar. 1994

[Horaud89] Radu Horaud, Bernard Conio, and Olivier Leboulleux, An Analytic Solution for the Perspective 4-Point Problem, Computer Vision, Graphics, and Image Processing 47, 1989, pp. 33-44

[Hunt95] B.R. Hunt, Superresolution of Images: Algorithms, Principles, Performance, IJIST(6), No. 4, 1995, pp. 297-304

[Huttenlocher90] D.P. Huttenlocher and S. Ullman, Recognizing Solid Objects by Alignment with an Image, IJCV, No. 2, November 1990, pp. 195-212

[Kender76] J. Kender, Saturation, hue, and normalized color: Calculation, digitization effects, and use, CMU Computer Science Dept., Nov. 1976

[Koller97] Dieter Koller , Gudrun Klinker, Eric Rose, David Breen, Ross Whitaker, and Mihran Tuceryan, Real-time Vision-Based Camera Tracking for Augmented Reality Applications, To appear in the Proceedings of the Symposium on Virtual Reality Software and Technology (VRST-97), Lausanne, Switzerland, September 15-17, 1997

[Kutulakos96] K. N. Kutulakos and J. Vallino, Affine Object Representations for Calibration-free Augmented Reality, Proceedings of IEEE Virtual Reality Annual International Symposium. 1996

[Lee97] S. Lee and S. Ro, Uncertainty Self-Management with Perception Net Based geometric Data Fusion
, ICRA, 1997

[Linnainmaa88] Seppo Linnainmaa, David Harwood, and Larry S. Davis, Pose Determination of a Three-Dimensional Object Using Triangle Pairs, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 10, No. 5, September 1988, pp. 634-647

[Madritsch96] F. Madritsch, F. Leberl, and M. Gervautz, Camera based Beacon Tracking: Accuracy and Applications, Proceedings of the ACM Symposium on Virtual Reality Software and Technology, pp. 101-108, July 1996

[Mellor95] J. P. Mellor, Enhanced Reality Visualization in a Surgical Environment, AI Technical Report 1544, 1995

[Neumann96] Ulrich Neumann and Youngkwan Cho, A Self-Tracking Augmented Reality System, Proceedings of the ACM Symposium on Virtual Reality Software and Technology, pp. 109-115, July 1996

[Nevatia77] R. Nevatia, A color edge detector and its use in scene segmentation, IEEE Transactions on System, Man, and Cybernetics, vol. SMC-7, pp. 35-37

[Oakley91] J. P. Oakley and R.T. Shann, Efficient Method for Finding the Position of Object Boundaries to Sub-Pixel Precision, IVC(9), 1991, pp. 262-272

[State96] Andrei State, Gentaro Hirota, Davod T. Chen, Bill Garrett, and Mark Livingston, Superior Augmented Reality Registration by Integrating Landmark Tracking and Magnetic Tracking, SIGGRAPH 1996, pp. 429-438

[Sung92] Kah-Kay Sung, A Vector Signal Processing Approach to Color, Master thesis, MIT, 1992

[Tuceryan95] M. Tuceryan, D. S. Greer, P. T. Whitaker, D. Breen, C. Crampton, E. Rose, K. H. Ahlers, Calibration Requirements and Procedures for Augmented Reality,

[Ward92] M. Ward, R. Azuma, R. Bennett, S. Gottschalk, and H. Fuchs, A demonstrated optical tracker with scaleable work area for head-mounted display systems, Proceedings of 1992 Symposium on Interactive 3D Graphics, 1992, pp. 43-52