What if cameras could see themselves?

D. Weinshall‡ M.S. Lee† E. Cohen-Solal† A. Colmenarez†

‡School of Computer Science
Hebrew University of Jerusalem
91904 Jerusalem, Israel
daphna@cs.huji.ac.il

† Philips Research
345 Scarborough Road
Briarcliff manor, NY 10510, USA
Mi-Suen.Lee@philips.com

Abstract

We propose a camera design where the camera records images through an attached aperture, which can be seen in the image. In such a design camera calibration is reduced to the problem of camera localization, with 3 degrees of freedom remaining which correspond to the unknown location of the camera. 9 degrees of freedom which correspond to the unknown camera's internal parameters and orientation are eliminated by the registration of the aperture. We discuss two applications: pointing target detection when pointing towards the camera, and depth estimation from two or more uncalibrated cameras. We conclude with experimental results which demonstrate the usefulness and robustness of our approach.

1 Introduction

In the absence of prior knowledge about the camera, including its internal calibration and its orientation and position in the scene, the extraction of metric 3-D information about the environment can be very challenging. One of the sources of difficulty is that under ideal conditions, the camera's internal and external parameters introduce 12 free parameters whose value is unknown. In this paper we focus on this problem, and show how we can reduce the number of free parameters to 3 by having the camera view the world through a permanently attached and visible aperture. In this setup, the only uncertainty that remains comes from the unknown position of the camera; the camera's orientation and internal calibration are automatically cancelled out, and do not affect subsequent computations.

The distinguishing feature of our approach is the idea to replace traditional ways of camera calibration, which compute pose and internal parameters explicitly, by plane registration and camera localization. Plane registration is a 2-D transformation (homography) that can be computed efficiently and robustly from image measurements alone; such plane registration of the visible aperture removes the dependency of the image measurements on the orientation of the camera and its internal parameters. Our approach is flexible in that the cameras may change their internal and external parameters from frame to frame in an unconstrained way, while the scene may also change (it is not assumed to be rigid). We outline the relation between our work and previous work in section 2.

In Section 3 we describe the details of our approach. Specifically, we describe the new design and show how camera calibration is reduced to camera localization, in which the relative position of the camera's focal center with respect to the aperture plane should be computed. Unlike unconstrained camera calibration, this computation is well-conditioned and can even be solved by linear methods. In order to demonstrate the usefulness of our approach, we applied it to two very different problem domains. The first is gesture recognition, discussed in Section 4. The second application, metric 3-D reconstruction from uncalibrated cameras, is discussed in Section 5.

Our approach agrees with a certain uneasiness about camera calibration as expressed, for example, in [2]. The basic concern is that unconstrained camera calibration can be very sensitive to small changes in the image data, while the reconstructed structure appears to be less so. If the purpose of the computation is structure reconstruction and not camera calibration per se (as is typically the case), it seems unjustified to put too much effort into precise camera calibration. Our approach is in line with this notion: we try to eliminate the dependency on camera parameters as much as possible using image registration, recovering (if necessary) only a small number of camera-specific parameters.

To motivate our approach we explored two applications, the first of which is pointing gesture recognition. Pointing at planar surfaces such as TV and computer monitors or projection screens can be a useful mode of interaction between humans and machines, e.g., for choosing items from a menu or playing a computer game. Identifying the pointing target is particularly challenging when the target plane is not in view of the camera, such as the case when the camera is mounted on the computer monitor. We use our approach to address this problem in Section 4.

2 Relation to previous work

2.1 Camera calibration

Previous work on camera calibration can often be categorized as one of the following:

1. The traditional approach uses a calibration object - a known 3-D object, typically including 2 or more orthogonal planes marked by a precise grid of fiducial points (or a plane undergoing known translation); the different appearance of the calibration object in the two images can be used to calibrate the two cameras. This approach was adopted early on by the photogrammetry community; its main advantage is the potential for high precision and robust reconstruction. Its drawbacks include: expert assistance - the calibration process is typically tedious, requiring an expert operator, and the availability of a very precise calibration pattern (but see [26] for a less tedious method); inflexibility - once the cameras are calibrated they cannot be moved and internal parameters cannot be changed; high cost - this approach typically requires an elaborate setup and expensive calibration equipment, while imposing high computational load since solving calibration exactly involves non-linear optimization.

2. The second approach is often called self-calibration (e.g., [17]): the basic idea is to exploit a sequence of images (at least 3), and some constraints on the change in camera parameters (e.g., orthographic projection, and/or that only a few internal parameters ad changed along the sequence), to compute the camera parameters. A few recent algorithms have been developed along these lines, showing promising results (e.g., [16]). The main advantage of this approach is its generality - it can be used with any image sequence that obeys the basic assumptions. Its main drawbacks are the underlying assumptions such as scene rigidity. The second problem concerns the need to be given a sequence of images (thus a pair of images is not enough) taken with essentially the same camera whose parameters may change only to a limited extent; in fact, robust results require that the changes in internal camera parameters are limited to a few known parameters. These limitations make the approach unsuitable for depth from stereo, when a single image pair is taken with two cameras in general position and with unknown internal parameters.

Euclidean reconstruction from a sequence of uncalibrated cameras has been discussed in various papers, see e.g. [9, 11]. Metric self-calibration of a stereo rig without a calibration object (but assuming scene rigidity) has been accomplished by moving the stereo rig and taking a few image pairs [17, 16], by performing pure rotation [21], or by assuming planar motion [1] (the lists of references above is only representative). Such active approaches suffer from the main drawback of the first approach above, namely, that calibration becomes an expensive initial process, after which the cameras cannot change their internal or external parameters.

Ours is essentially a hybrid approach which inherits many of the benefits of the two approaches described above, while avoiding the main drawbacks. It is a descendant of the traditional approach in that we use a known calibration object. However, our calibration object is part of the camera rig itself. This gives our approach its flexibility, namely, that the scene does not have to be rigid and the cameras may change in an unconstrained way from frame to frame.

Replacing camera calibration by plane registration + localization was inspired by the so called "plane + parallax" approaches (e.g., [15, 20]). It was first suggested as an alternative way of camera calibration in [23], using a scene plane as the reference plane.

2.2 Stereo

A robot carrying its own calibration object was described in [25]. In this stereo setup some common parts of the calibration object must be seen by both cameras simultaneously (as in the traditional calibration approach), which leads to a rather cumbersome setup where the stereo rig carries a large object at quite a distance in its front. In the stereo flavor of our setup we simply put a plane with two apertures on the tray on which the cameras are mounted, for example, and only require that each camera will have one of the apertures in its field of view. Our system is therefore small and practical, being able to move around without hitting anything and with minimal obstruction of the field of view.

2.3 Pointing

Previous work on pointing usually involves geometrical constructions in 3-D, resulting in high computational load and imprecise results. This kind of computation may be needed when the pointing target lies in an arbitrary position in space. Accordingly, the 3-D pointing vector is computed in [12, 14] using a stereo pair and in [7, 19] using two cameras located above and to the side of the subject. 2-D image-based reasoning is used in [13] where the pointing direction is identified in a single image; this system, however, does not locate the pointing target but only identifies a cone in the image which contains it. In [3], the pointing target on a ground plane is computed from two images generated by two uncalibrated stereo cameras, using only two-dimensional collineations.

3 A self-calibrating camera

One of the distinguishing features of our approach is the camera design, where the camera is attached to an aperture which is visible to the camera at all times. The aperture becomes, to a certain degree, the calibration object. In the stereo flavor of our approach, two cameras lie behind a plane which has two polygonal apertures cut side by side (see Fig. 1), and the cameras view the scene from behind the two apertures.

We call the plane which contains the aperture (or apertures) the *reference plane*. The reference plane and an orthogonal axis define a 3D coordinate system, which we call the *3-D reference coordinate system* (see Fig. 1). All 3D reasoning discussed below is done in this coordinate system; note, however, that this coordinate system changes with the camera as it moves. Initially and only once, a frontal image of the reference plane (screen + attached apertures) is taken at a very narrow field of view (approximately orthographic projection), to represent the $2D$ undistorted appearance of the reference plane; this image is our *reference frame*. A geometrically accurate schematic diagram of the reference plane may also be used.

Before any further processing, we register the visible portions of the aperture in each image with the *reference frame*. This relatively simple operation involves two steps: the correspondence of a few aperture features between the image and the *reference frame*, and the computation of the $2D$ linear transformation which registers the image with the *reference frame*. As illustrated in Fig. 1, we could transform each image so that the four image points $p1', p2', p3', p4'$, which are defined by the corners of the aperture, are mapped to the positions of the corresponding reference plane points $p1, p2, p3, p4$. The fact that we map a plane (the

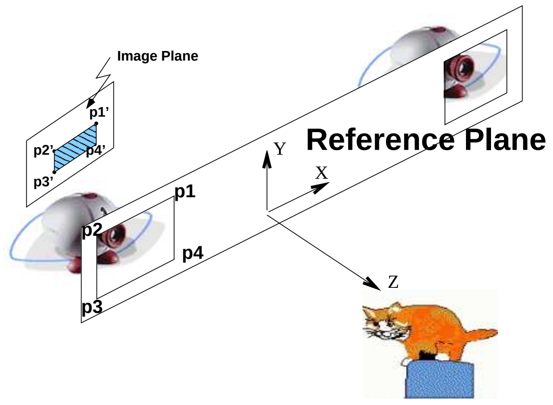


Figure 1: Graphic illustration of the reference plane (side view) in the stereo case: two apertures are cut in the plane through which the two cameras view the scene; the aperture as seen in the right camera is also shown.

image plane) to a plane (the reference plane) guarantees that this can be done with a projective $2D$ linear transformation (homography).

It is shown in the appendix (see also [23]) that after registration, the image generation via projection performed by each camera (assumed to be an ideal pinhole camera) is defined by the following simple equation:

$$\mathbf{p_t} \propto \begin{bmatrix} -Z_t & 0 & X_t & 0 \\ 0 & -Z_t & Y_t & 0 \\ 0 & 0 & W_t & -Z_t \end{bmatrix} \mathbf{P} \tag{1}$$

where $\mathbf{p_t}$ denotes the location of a point in the registered image, $\mathbf{P}$ denotes the location of the same point in the *3-D reference coordinate system*, and $[X_t, Y_t, Z_t, W_t]$ denote the location of the focal center of the camera in the *3-D reference coordinates system*. All the quantities above are measured in homogeneous coordinates. Note that the matrix operator in (1), which describes the projection from the 3-D world to the 2-D image, does **not** depend on the camera's internal parameters or orientation, only on the location of the camera's focal center. This is a unique feature of the registration process adopted here, and one of the main reasons to use it.

After registration, the only uncertainty that remains in the image is due to the unknown location of the camera, total of 3 degrees of freedom. In the applications discussed below we treat this uncertainty in two ways:

Pointing identification: when the task is to identify a pointing target on the *reference plane*, the target can be uniquely identified by the intersection of image lines.

Depth computation: in order to compute absolute depth relative to the *reference plane*, we compute the location of the camera' focal center in the *3-D reference coordinate system*. The computation uses Eq. (1) and two known calibration markers that lie outside the reference plane.

4 Application I: gesture recognition

In this application the task is to identify the target of pointing on a flat surface such as a computer monitor. The adaptation of the above design to this task is shown in Fig. 2. Specifically, a vertical opaque plane with one or two polygonal apertures (aligned side by side) is positioned on top of the computer or TV

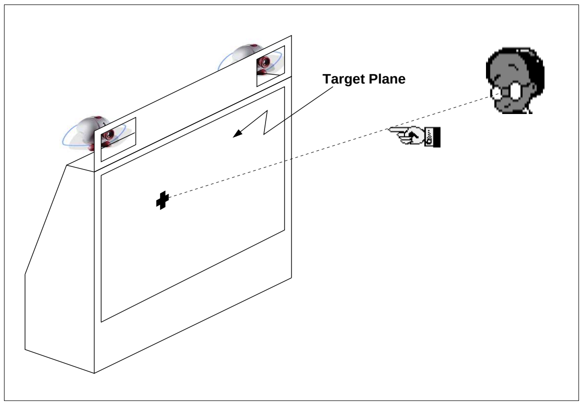


Figure 2: Graphic illustration of the target plane (side view): the target plane is a TV or computer monitor, with a planar extension added on top; two apertures are cut in the extension plane, through which the two cameras view the scene.

monitor, approximately coplanar with the screen. The aperture is always fully visible in the respective image periphery. The extended plane, which includes the screen and the extension containing the aperture (or apertures), is called the *target plane*. The *reference frame* as defined above is a picture of this *target plane*. In this design the user is photographed pointing towards the camera; to find the pointing target, it is necessary to detect the user's eyes and finger-tip.

Pointing Target Detection in our approach involves the following steps:

Target plane registration: Identify corresponding points in the image and the target plane, and compute the 2-D linear transformation which maps the image points to the target plane points. This transformation is the *registration homography.*

Pointing direction: In each original image, detect the pupil of the pointing eye and the tip of the pointing finger (which may wear a detectable thimble). Apply the *registration homography* to these points, and compute the line passing through the two points. This line defines the pointing direction, and provides a linear constraint on the location of the pointing target in the target plane.

Target detection: Using two or more linear constraints, identify the exact location of the pointing target.

These steps are discussed at greater length below in Section 4.1, while issues of implementation are discussed in Section 4.2. Examples are shown next: In the first experiment (Section 4.3) two cameras provide two constraint lines, whose intersection is the pointing target. In the second experiment (Section 4.4) one camera provides one constraint line, while the second constraint line is given a-priori (for example, the target may be constrained to lie on a horizontal menu). We note that most subjects showed consistent bias in their pointing (either to the right or above the target) in the first experiment. Hypothesizing that this might be due to the large size of the color thimble used in this experiment, we used a smaller flicker-light thimble in the second experiment.

4.1 Image constraints on the pointing target

The geometry underlying the method is described in Figs. 3a,b. Fig. 3a shows the setup: the imaginary line going between the person's eye and finger-tip intersects the target plane in the pointing target precisely (but see discussion in Section 4.2.4). Assuming for the moment that the target plane is the image plane, the pointing direction is projected onto the image through the camera's focal point (Fig. 3b); it is the line going through the person's eye and finger-tip as seen in the image, and it passes through the pointing target. The essence of our method is to use this line in order to compute the pointing target directly in the image plane.

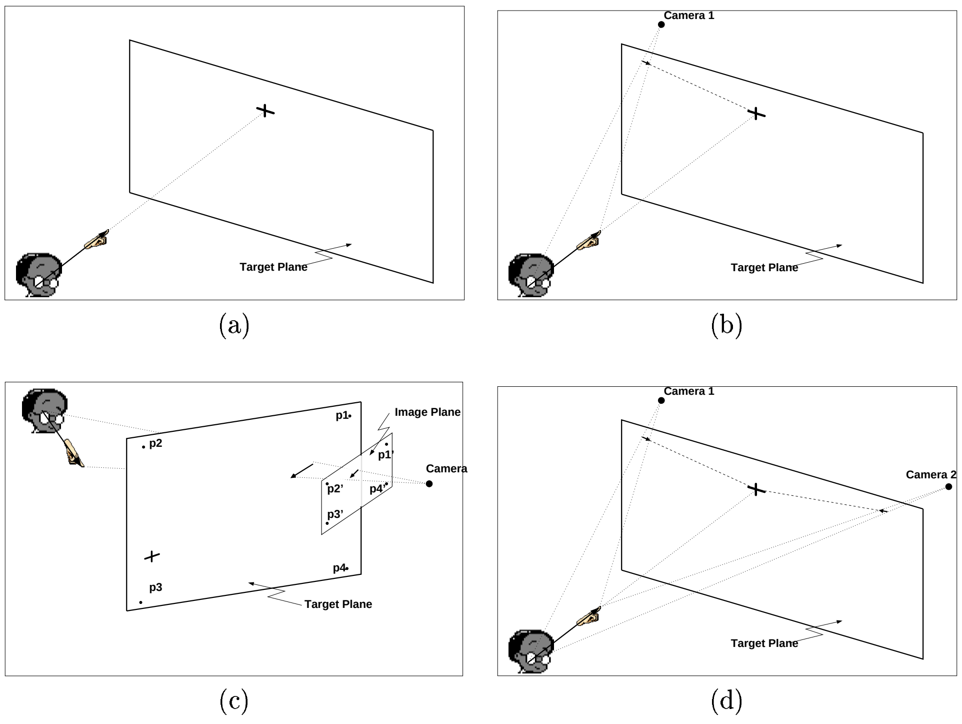


Figure 3: (a) A person pointing towards a flat surface: the three-dimensional line going between the person's eye and fingertip intersects the target plane in the pointing target. (b) The pointing direction is projected on the target plane through the camera's focal point **Camera 1**, and it passes through the pointing target. (c) Morphing: using 4 corresponding points the image is morphed to agree with the target plane. (d) Using 2 cameras we get 2 different lines on the target plane whose intersection identifies the pointing target.

In reality the target plane is not the image plane, as shown in Fig. 3c. However, if the target plane has been registered with the image plane prior to the computation, for all practical purposes we can consider the two planes to be identical. In the example shown in Fig. 3c, we transform the image so that the four image points $p1', p2', p3', p4'$ are mapped to the positions of the corresponding target plane points $p1, p2, p3, p4$ (via a homography).

In order to detect the location of the target in the image, we need at least two linear constraints on the point. Fig. 3d illustrates the essence of our method in the general case: two cameras provide two different lines (or two linear constraints) on the target plane which pass through the pointing target, and thus their intersection identifies the pointing target. If the pointing target is constrained to lie on a line on the target plane, e.g. if it must lie on a horizontal menu, then one linear constraint from one camera is sufficient.

4.2 Implementation details

We now provide the details of how exactly we compute the registration homography, the image pointing direction, and how we detect the pointing target.

4.2.1 Image based registration

We identify in each image points on the aperture boundaries, which by construction lie on the target plane. Typically these are the corner points which can be accurately identified as the intersections of the four lines bounding the aperture [8] (see example in Fig. 4). During initialization these corners are automatically detected on the *reference frame* using the same line intersection procedure. We then compute the 2-D homography which moves the image points to their veridical position on the *reference frame*. To compute the registration homography we need at least 4 points, or at least 2 pairs of parallel lines.

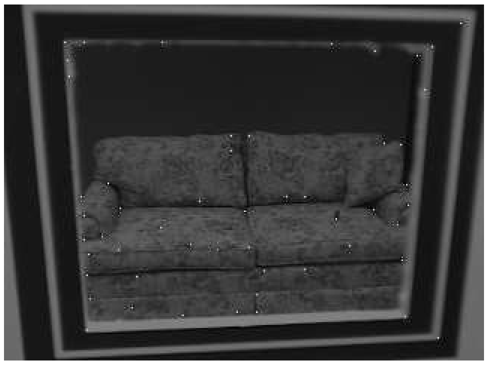


Figure 4: The output of the Harris corner detector [8] on one image. The features used in our experiments are the corners of the internal frame in the aperture.

More specifically, let (x, y) denote the image coordinates of a point, and (X, Y) denote its coordinates on the *reference frame*. The registration transformation is a 3×3 matrix $\mathbf{M}$ which transforms one representation to the other in homogeneous coordinates:

$$\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} \propto \mathbf{M} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

where $\propto$ denotes equality up to scale. A simple counting argument shows that since each point provides two constraints on $\mathbf{M}$, and since $\mathbf{M}$ is defined up to scale (thus containing 8 unknowns), at least 4 points (or 2 parallel lines) are needed to compute the transformation.

4.2.2 Eye detection

We locate the face and the corresponding eye corners in each image using an algorithm for face and facial feature detection and tracking based on Information-Based Maximum Discrimination (IBMD) classifiers [4]. Tracking is done in the unwarped images. In this algorithm, discrete, non-parametric probability models are used in a maximum likelihood setup. The observation model for each face and facial feature consists of the combination of three distinct low-level image features: pixel intensity, vertical edges, and horizontal edges.

The required probability models of faces and facial features are obtained from both positive and negative examples. The IBMD training procedure produces models that maximize the information-theoretic discrimination (e.g. Kullback-Leibler divergence) between the likelihood of the positive and negative classes. The eye corner detection program used was trained using face images from the FERET database. Positive examples were obtained from image regions at the given ground-truth location of the eye corners in these images; negative examples were obtained from the surrounding image windows.

Face and eye corner detection is achieved by locating the positions in the tested images with the highest likelihood of observation (Fig. 5a). Tracking consists of updating these positions by maximizing the likelihood of observation within a neighborhood of the previously known location of each feature. Face detection is carried out at a very low resolution, while facial features are located at double resolution. On a given image, the algorithm can detect faces as small as the size of the face model – sub windows of 16×14 pixels. Larger faces are detected by searching over a multi-scale pyramid of images. In this implementation, we also used scale factors greater that one (image interpolation by a factor of 2) to allow the program to locate faces at half the size of that of the face and facial feature models.

After this automatic detection of faces and eye corners, we estimate the location of the right pupil as the point lying on the line connecting the left and right eye corners, roughly 15% away from the right corner towards the left corner. The left pupil is estimated in a similar way, relative to the left eye corner (see Fig. 5a).

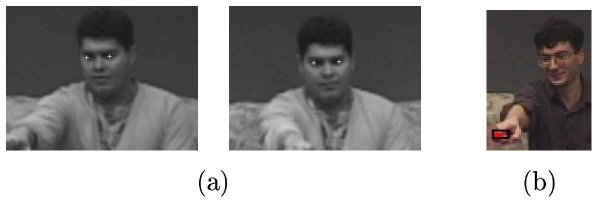


Figure 5: (a) The output of the eye corners detector on one example of a face stereo pair. (b) Output of finger tip detector: the tracked area around the pointing finger tip is marked red, highlighted by a rectangular frame.

4.2.3 Finger tip tracking

In this set-up the user typically points to a target located in relative proximity to the cameras themselves, which makes the finger tip detection task quite difficult: the finger-tip tends to occlude the torso, which makes detection based on temporal changes very hard. To simplify the detection and establish "control", the user wears an easy to detect thimble, identified either by a large color blob or by intense flicker light. The advantage of using high intensity flicker-light is that it generates a large detectable blob in the image (due to saturation of nearby pixels), while maintaining a small actual size which allows people to aim their finger accurately. Tracking and detection are done in the unwarped images.

In the first example below (Section 4.3) the identifying feature of the thimble is color. To detect the thimble we first initialize the color model by training the system with the colored thimble. We use a color model in the HSI color-space, and Gaussian approximation for color distribution. Our tracker is based on the combination of color pixel classification and local search in the next frame based on previous detection. We estimate the finger tip as the centroid of the image region identified as the thimble (Fig. 5b).

In the second example below (Section 4.4) the identifying feature of the thimble is high intensity flicker light. It is tracked in real time by searching for pixels that exhibit a regular, pulsating temporal pattern. Due to the small size of the thimble (approximately the size of an actual finger-tip), the results of finger-tip detection are very accurate in this design.

Previous work on finger tip detection include the application of a new image transform to locate finger tip [22], the use of neural networks to locate 2-D finger tips and generate a 3-D model for synthetic visualization [18], and index tip detection to point to a room on a building map. All of these papers have in common that the camera is looking perpendicularly to the arm/hand/finger to perform a 2-D location of the index tip. Recently, Wu et al [24] performed 3-D finger tracking using skin detection and arm motion tracking to find the finger. Pointing was not discussed in these applications.

4.2.4 How people point

To compute the pointing direction, we first need to know how people define it. In our informal experiments with three adults and one toddler, we asked subjects to point towards a large target screen at a distance of roughly 2 meters. The experiments show that people define rather precisely a pointing direction from the right or (less often) left pupil through the tip of the pointing finger (Fig. 6). This was verified by drawing a line between the pointing target and the finger-tip; as illustrated in Fig. 6, this line intersects rather precisely the right eye of the subject. A weaker observation was made in [7, 13], where it was noted that people use the line between their head and the finger tip as the pointing direction.

4.2.5 Potential difficulties

There are a few sources of errors and uncertainties in the proposed method:

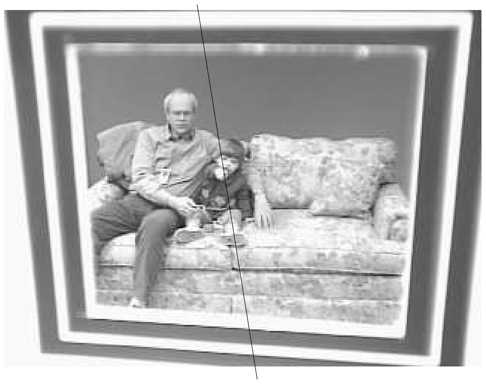


Figure 6: Snapshots from our informal experiments: the line connecting the pointing target (assumed known) and the finger-tip of the subject intersects the right eye of the subject.

1. Due to personal variability, not all subjects may be able to articulate the pointing motion as intended (for example, due to poor vision or weakness of the pointing arm).
2. The localization of eye and finger tip is susceptible to uncertainty, sometimes in the order of magnitude of a few pixels. Such uncertainty in point localization translates to an uncertainty cone around the estimated line. The shorter the distance between the two points defining the line is, the larger the cone of uncertainty is.
3. As the linear transformation is a 2-D homography, small errors in point localization can lead to large errors in the computed transformation if the perspective effects are strong.

4.3 Experimental results: two cameras

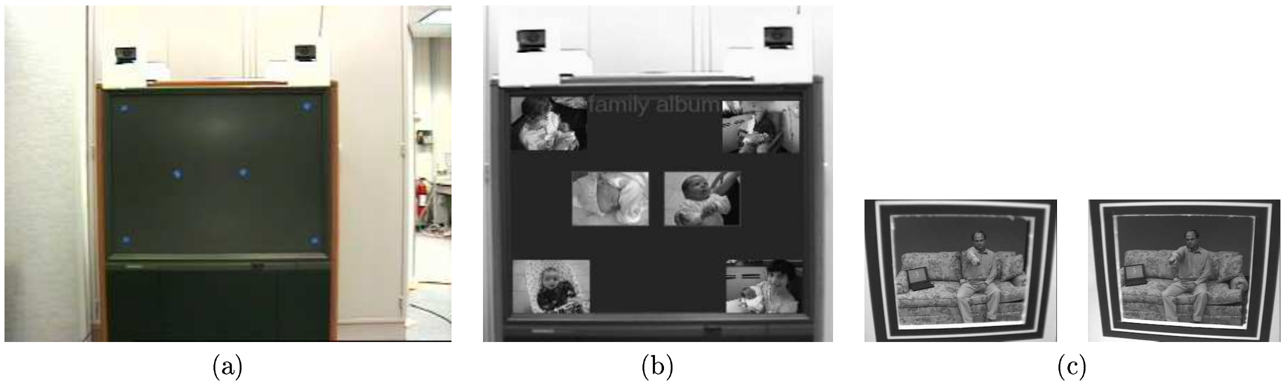


Figure 7: a) The *reference frame*, a fronto-parallel small field picture of the target plane (the TV screen) and the attached apertures, as used in our experiments. The six pointing target points used in our experiments were marked on the TV screen with blue bullets. b) An illustration of a photo album sorting application, where pointing is used to select pictures. c) A pair of frames (left and right cameras) from the actual stereo cameras.

Our experiments were conducted in a laboratory, with subjects sitting on a couch and pointing towards a 46″ TV monitor standing 2.5 meters away from the couch. Two cameras stood on the monitor viewing the scene through apertures cut in a plane roughly coplanar with the monitor. The distance between the cameras was roughly $80cm$, and the size of each aperture was roughly 9×9.5 cm. The *reference frame*, a

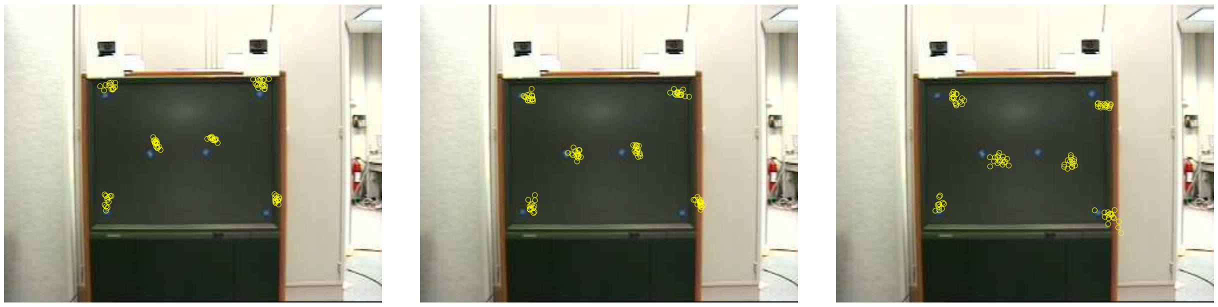


Figure 8: Pointing results for three subjects. Similar results were obtained for a fourth subject. Two sequences, showing experimental sessions and the corresponding identification results, are available in mpeg from our web site http://www.huji.ac.il/~daphna/demos.html#pointing. In these movies, the projection of the pointing direction line onto the reference plane via each camera is shown in red when the subject is in transition, and in green when stable pointing is taking place.

frontal view of the TV monitor and the two apertures, is shown in Fig. 7a. An example of an application is shown in Fig. 7b.

In this experiment three subjects were asked to points towards one of six markers on the screen (Fig. 7a) in sequential order; beeps at 5 seconds intervals marked the initiation of each step. One example stereo pair from this experiment is shown in Fig. 7c.

For each subject and each frame, we draw in Fig. 8 a yellow circle around the pointing target detected as described above. We eliminated transitional frames, taken immediately after the beep sound. when subjects moved their finger to point at a new target. For all the subjects the data clustered neatly close to the six intended pointing targets, showing high relative precision in both the subjects' pointing and the target detection. The median errors, computed as the distance between the intended target and the identified target, were 6.2 *cm*, 6.6 *cm*, and 9.85 *cm* for the three subjects respectively. Given that the subjects were sitting at 2.5m away, the median angular error is less than 0.04 radians. Table 1 summarizes the results.

subject	median error	min error	max error
1	3.5 -5.1	-0.103 -0.3248	7.6 -8.3
2	6.6 -0.1	-0.286 -0.0005	-11.9 -7.6
3	9.1 3.7	0.304 0.0400	20.6 10.0

Table 1: Identification errors measured in cm: the median error, minmal error and maximal error are shown for each subject and for each direction (horizontal and vertical)

4.4 Experimental results: one camera

This experiment was conducted using a similar setup to Section 4.3, with one important difference: only one camera stood on the monitor (roughly centered), viewing the scene through an aperture cut in a plane roughly coplanar with the monitor. The target plane, designed in a way suitable for children's games, is shown in Fig. 9. The target points lie along two one-dimensional curves shaped roughly like half-circle. We used two different menus, each composed of a set of target points aligned over a different curve: 10 target points on the first menu (upper curve), and 6 on the second menu (lower curve). Now the intersection of the single image pointing direction and the curve along which the targets lie uniquely determines the pointing target.

For each subject and each frame, we draw in Fig. 10 a dark line, showing the computed pointing direction registered with the reference frame as described above. Here, too, we eliminated transitional frames, when

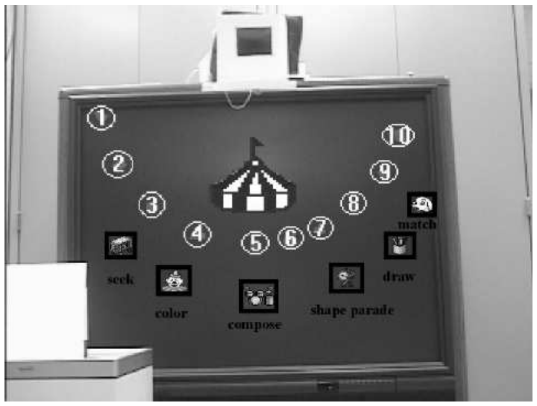


Figure 9: The *reference frame*, a fronto-parallel small field picture of the target plane (the TV screen) and the attached aperture, as used in our pointing experiment with one camera. The target points are marked: 10 encapsulated numbers on an upper curve and 6 icons on a lower curve. This picture demonstrates the use of pointing in children's games, where pointing is used to select levels and games from two menus.

subjects moved their finger to point to a new target. For all the subjects the lines cluster together neatly in the proximity of the intended pointing target in the vicinity of the constraint line (the menu curve). The results show high relative precision in both the subjects' pointing and the target detection.

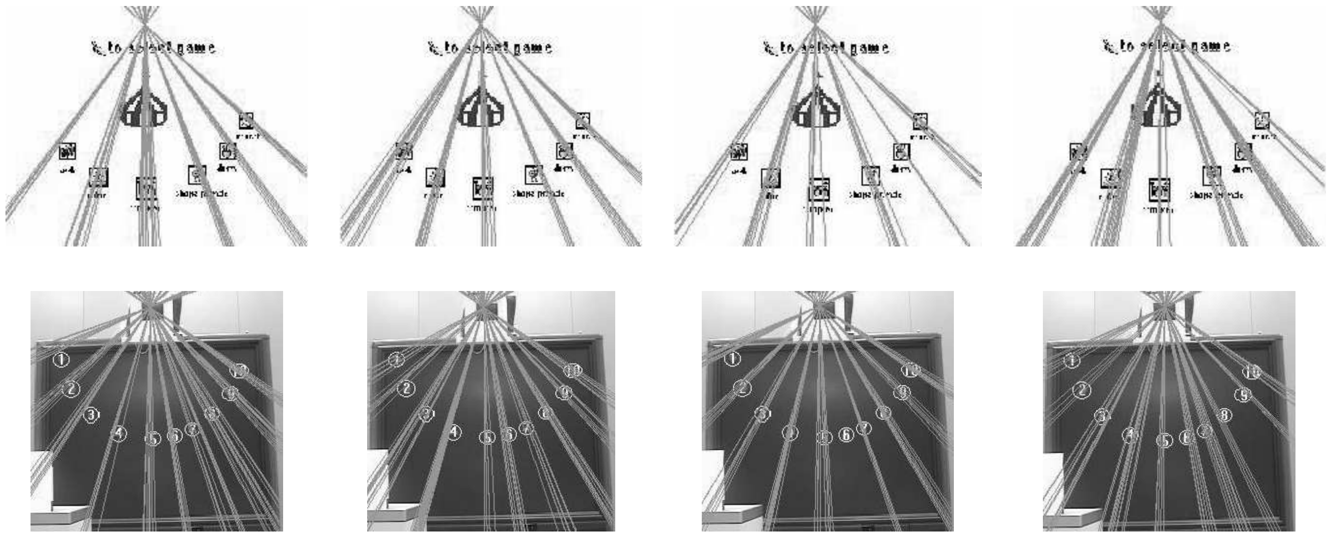
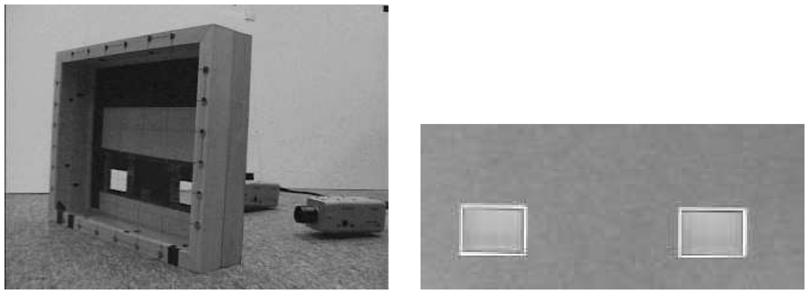


Figure 10: Pointing results for 4 subjects, using one camera; results are shown for MSL, MIN, MDT and Dan, from left to right respectively. The upper row shows pointing results at the lower (game selection) menu, superimposed on the original game board. The lower row shows pointing results at the upper menu superimposed on the level selection menu.

4.5 Discussion

Our system requirements, i.e. the fact that we use un-calibrated cameras arbitrarily positioned, can be met in inexpensive consumer applications operated by non-expert users. In comparison, most existing approaches which use 3-D pointing detection can only be implemented in very specialized setups, with calibrated cameras and expert users who can maintain the cameras calibrated.



a) b)

Figure 11: a) The experimental setup: two cameras view the scene from behind an opaque plane (the reference plane) through two apertures cut in the plane. A grid is stationed in front of the reference plane, at depth of roughly $7.5cm$, so that each camera can see a few grid points through the aperture in its field of view. b) Approximately orthographic picture of the reference plane, to be used as the reference frame.

An obvious extension of our work is to provide feedback to the user so that target identification errors can be corrected interactively. However, in a few preliminary experiments, we observed that when real-time feedback is given to the user, the initial position of the target is much less important than the responsiveness of the feedback loop. In light of these observations, we argue that our method is most suitable for applications where feedback is not desirable, e.g., a game where each player chooses one of a few possible answers or moves without revealing their intentions to the other players. When feedback is possible, e.g., applications that require item selection such as in traditional TV and computer user interfaces, our method can still be used but without much emphasis on accuracy (thus one camera is almost certainly sufficient).

We note that to aid in the detection of the finger-tip, the user is asked to wear a thimble; the thimble replaces other active means of interaction, such as a remote control. In addition to making finger-tip detection easier, the thimble has the following advantages for user-interface purposes: (i) it tells the system which user has "control" at a given time; (ii) if there is more than one user, as in the context of a computer game with a few players, the characteristics of the thimble provide user ID.

5 Application II: depth from stereo

In this application the goal is to compute relative depth from dynamic uncalibrated stereo cameras. An instantiation of the stereo rig, as used in our experiments, is shown in Fig. 11a. As discussed in Section 3, the self-calibrating rig calibration involves two steps:

Image registration: Identify corresponding points in the first image and the reference plane (e.g., the aperture corners), and compute the homography which maps the image points to the reference plane. This step is identical to the procedure described in Section 4.2.1. Repeat the same procedure over the second image.

Localization: In each image identify the projection of two known points on the calibration object (the stereo rig) which are **not** on the reference plane. Apply the *registration homography* to the coordinates of the points, and use them to compute the location of the camera's focal center using (1) (see details in Section 5.1.1). Each camera may use different points at this stage as well.

Once calibration is known, the exact location of every 3-D point can be computed as follows:

1. Compute the fundamental matrix from the calibrated cameras (see details in Section 5.1.2).

2. Find image correspondence using the fundamental matrix to define the epipolar lines.
3. Use triangulation to compute the 3-D coordinates in the reference coordinate system of each point whose correspondence is known (see details in Section 5.1.3).

5.1 Implementation details

5.1.1 Localization

After registration, the projection from 3-D to 2-D as performed by each pinhole camera is described by Eq. (1). This equation has 4 homogeneous unknowns describing the location of the camera's focal center in the *3-D reference coordinate system*: $[X_t, Y_t, Z_t, W_t]$, which can only be computed up to scale. To compute each camera center we use two or more corners (as many as are visible through the aperture) on a grid located in front of the reference plane (see Fig. 11a). Since each grid point provides two constraints on our 4 homogeneous unknowns, we can solve for the camera center using 2 or more grid points.

5.1.2 Epipolar geometry

The fundamental matrix defines the epipolar geometry (for the purpose of correspondence) between the two images. More specifically, if $\mathbf{p}$ and $\mathbf{p}'$ denote respectively the homogeneous coordinates of a feature point in the two images, and $\mathcal{F}$ is the 3×3 fundamental matrix describing the two cameras' geometry, then the following holds [5]:

$$\mathbf{p}^T \cdot \mathcal{F} \cdot \mathbf{p}' = 0$$

This equation constrains $\mathbf{p}'$ (the matching point for $\mathbf{p}$) to lie on the epipolar line defined by $\mathbf{p}^T\mathcal{F}$.

After registration and localization, for $[X_t, Y_t, Z_t, W_t]$ and $[X_t', Y_t', Z_t', W_t']$ denoting the locations of the two cameras' centers respectively, it is shown in the appendix that:

$$\mathcal{F} = \begin{pmatrix} 0 & \frac{W_t}{Z_t} - \frac{W_t'}{Z_t'} & \frac{Y_t'}{Z_t'} - \frac{Y_t}{Z_t} \\ \frac{W_t'}{Z_t'} - \frac{W_t}{Z_t} & 0 & \frac{X_t}{Z_t} - \frac{X_t'}{Z_t'} \\ \frac{Y_t}{Z_t} - \frac{Y_t'}{Z_t'} & \frac{X_t'}{Z_t'} - \frac{X_t}{Z_t} & 0 \end{pmatrix} \tag{2}$$

We note that the direct computation of $\mathcal{F}$ from the above expression is very sensitive to small errors in the localization process.

Instead, we can compute the fundamental matrix using a variation on the 8 point algorithm [5, 10]. Specifically, since $\mathcal{F}$ in (2) has only 3 unknowns defined up to scale, in *the reference frame coordinate system* we only need two corresponding points. However, if there are no corresponding points between the two images, we can reproject the out-of-plane calibration points from each camera to the other one using Eq. (1); we may then use the actual and reprojected image positions of at least 4 corresponding points (2 from each camera) to compute $\mathcal{F}$. This procedure gave good results, as can be seen in the examples shown in Section 5.2. The fundamental matrix and the epipolar geometry can now be used to find pairs of corresponding points in the two images.

5.1.3 Triangulation

To reconstruct the 3-D coordinates of image points in the *3-D reference coordinate system* we use once again Eq. (1). Now the projection matrix is known and the 4 unknowns are the 3-D homogeneous coordinates of the point $\mathbf{P} = [X, Y, Z, W]$, which can be computed up to scale. Each image gives 2 homogeneous constraints on the 4 unknowns, and thus the corresponding image coordinates in two images are sufficient to solve for the unknown 3-D coordinates.

5.2 Experimental results

We built the setup shown in Fig. 11 and obtained a few stereo pairs, each taken under different and unknown camera configurations (i.e., different internal and external camera parameters). The implementation was done in Matlab, by finding MSE solution to each respective linear system. The reconstruction and all its internal functions took about 40 lines of Matlab code. The points used for plane registration in the first stage of the algorithm were extracted automatically, using the Harris corner detector [8]. 3-D feature points could typically be extracted automatically, with correspondence found using the reconstructed epipolar geometry and 1-D search (but since the focus of this paper is *not* the correspondence problem, this automation has not yet been implemented in the experiments described below).

Our method is distinguished by two features: the use of the stereo rig itself as a calibration object, and the registration+localization way we use to calibrate the cameras. The first distinguishing feature requires special attention, as instability may be introduced due to the short distance of the calibration object from the cameras. To get a better feel for this problem, we implemented a different reconstruction method which still uses registration+localization for camera calibration, but which uses a traditional calibration object. More specifically, it uses a plane seen by both cameras on a calibration object in the scene as a reference plane, and two non-coplanar feature points on the calibration object for the localization. We tested this method on the first scene below, with results provided in Table 3 and Fig. 14.

Our method assumes a perfect pinhole camera, while practical cameras typically have various lens and other distortions. We addressed this problem experimentally, repeating the analysis using the same data before and after cubic corrections of the radial distortion. The corrections lead to a small and hardly significant improvement in the results. The results provided below use the corrected data in Section 5.2.1, and the raw data in Section 5.2.2.

5.2.1 Scene 1: calibration box

Three image pairs of a calibration box were taken at different distances from the cameras, with the same camera configuration (same internal and external parameters): the 'near box' condition where the calibration box stood 20" away from the cameras, the intermediate condition at 46" away, and the 'far box' condition at 100" away. They were registered with the reference frame, as shown in Fig. 12.[1] Subsequently the fundamental matrix was computed following the procedure described in Section 5.1.2. The accurately recovered epipolar geometry is shown in Fig. 13.

No quantity of the environment was measured apriori, and in particular the distance of the box from the stereo rig was unknown. The quality of the reconstruction results was therefore measured by how accurately points on the box were reconstructed relative to an internal, object centered, coordinate system. More specifically, after reconstruction we transformed (by rotation and translation) the 3-D representation to agree as much as possible (in MSE sense) with the known geometry of the calibration box; this was done using QR factorization. Results are summarized below in Table 2, for 10 features points (a selection of corners on the calibration box).

Next we wanted to check whether the effective use of the camera rig itself as a substitute calibration object decreased robustness, due to the fact that the "calibration object" is now very close to the camera. We therefore implemented the method described above which uses registration + localization for camera calibration, but using instead the furthest box as an external calibration object. The results are summarized below in Table 3, demonstrating the superiority of our original method. Clearly the reconstruction results now for objects located far from the calibration object (e.g., the 'near box' pair) are far worse (see Table 3), and the fundamental matrix is wrong (see Fig. 14).

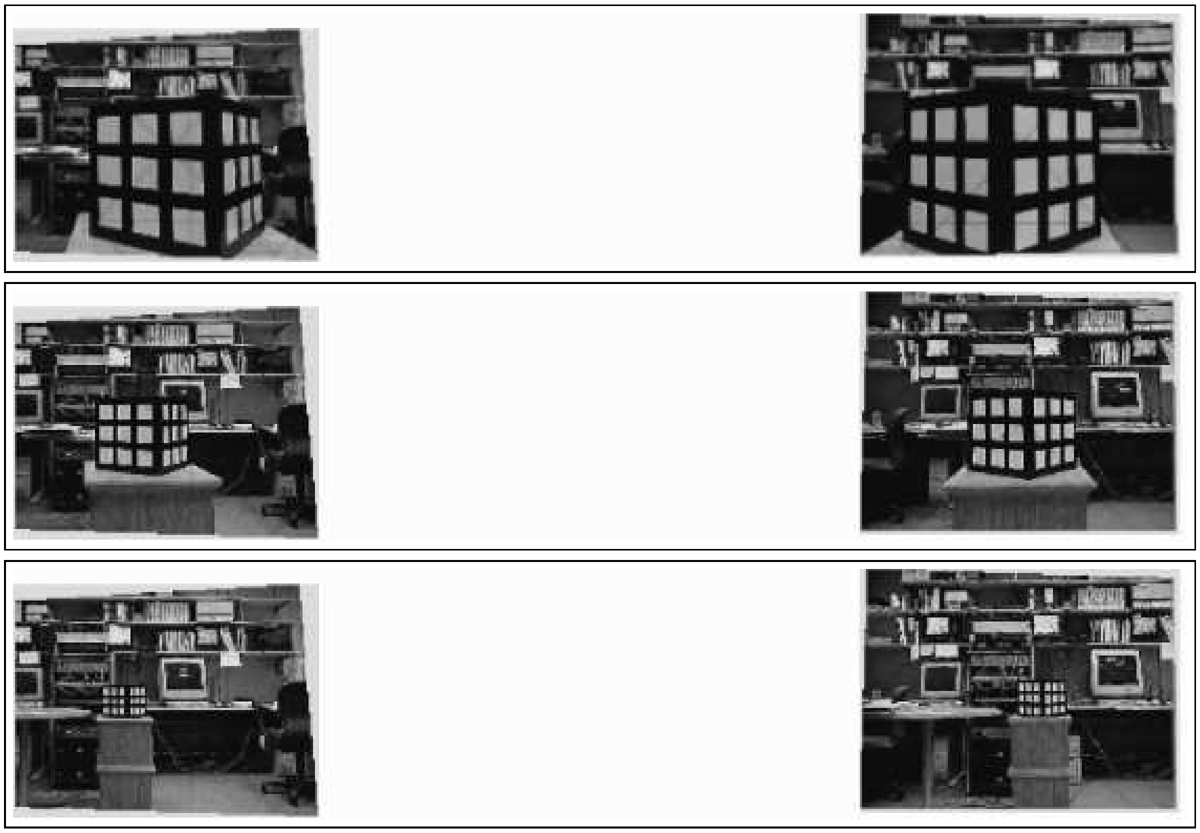


Figure 12: The three image pairs used in the calibration box experiment, registered with the apertures in the *reference frame*; the near, intermediate and far distance conditions are shown top to bottom respectively. The external frame of the *reference frame* is also shown in each case.

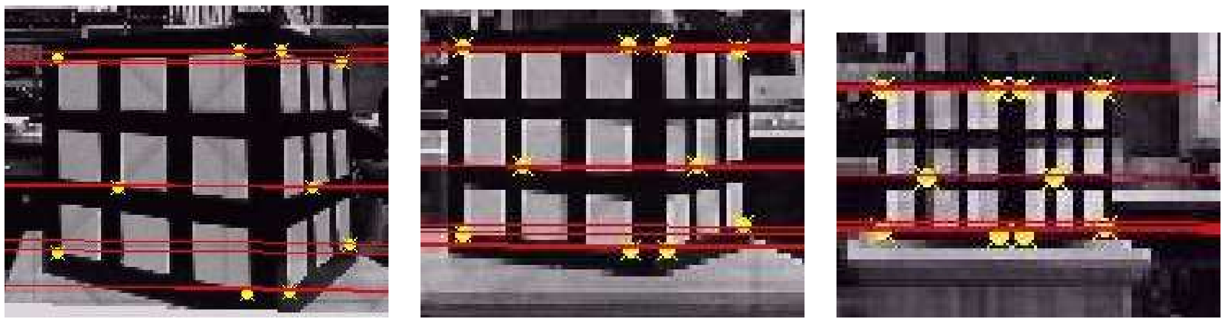


Figure 13: Epipolar geometry results, using our method: the epipolar lines computed for each image point (denoted by a circle+cross), as obtained from the computed fundamental matrix on the right image, are drawn over the left image. It can be seen that all of them pass very close to the matching left image point. Results are shown for the near, intermediate and far distance conditions from left to right respectively.

5.2.2 Scene 2: still life on the corridor floor

Three image pairs of an assortment of objects naturally lying on the corridor floor in our lab were taken, for three different camera configurations (with different and unknown internal and external parameters).

[1]In this experiment the reference plane did not have apertures but included a coarse grid, whose corners were used for registration instead of the aperture corners.

near box	mid-distance box	far box
(0.28, 0.12, 0.28)	(0.43, 0.29, 0.51)	(1.1, 0.73,1.3)
(1.2, 0.36)	(0.5, -0.03)	(0.41, -0.08)
(0.77, 0.46)	(0.5, -0.004)	(0.49, -0.07)

Table 2: Reconstruction results for the 3 image pairs used in the first experiment: The first line gives 3-D error, the mean square error in inches (for a calibration box whose size is 10 inches cube) of the 3-D reconstruction after optimal alignment with an object centered coordinate system (see text). The second and third lines give image reprojection error, the mean error in pixels between the reprojected image of the reconstructed box features and the actual coordinates of these features, respectively for the left and right images.

near box	mid-distance box	far box
(2.2, 0.94, 2.3)	(1.54, 0.49, 1.14)	(0.25, 0.05, 0.3)
(15.3, -3.13)	(2.73, -0.48)	(-0.05, -0.02)
(28.8, -9.6)	(3.64, -0.86)	(-0.07, -0.02)

Table 3: Reconstruction results for the three image pairs used in the first experiment with a traditional calibration object, see explanation in Table 2.

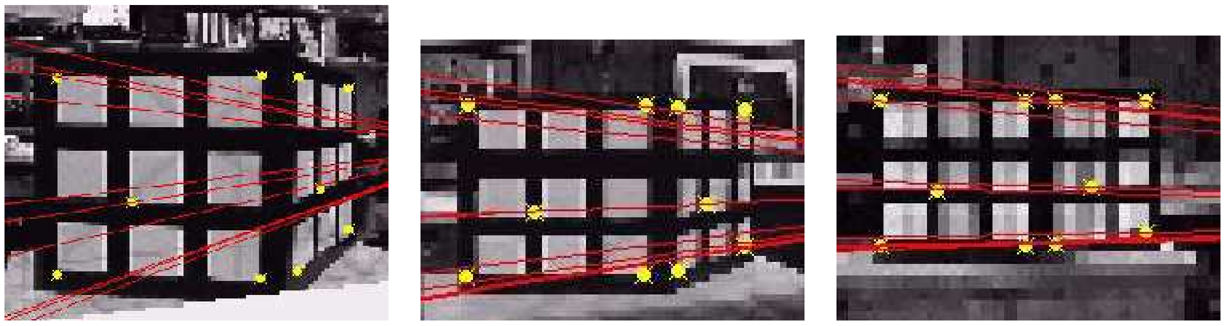


Figure 14: Epipolar geometry results, using our version of the "traditional" approach: the epipolar lines computed for each image point (denoted by a circle+cross), as obtained from the computed fundamental matrix on the right image, are drawn over the left image. It can be seen that in this case the epipolar lines do **not** pass close to the matching left image point, compare to Fig. 13.

They were registered with the reference frame, as shown in Fig. 15. The fundamental matrix was computed following the procedure described in Section 5.1.2. The accurately recovered epipolar geometry is shown in Fig. 16.

In these experiments the coordinates of each object in the *3-D reference coordinate system* are known, using the markings on the tiled floor. We computed the 3-D coordinates of each pair of matched feature points using the reconstruction algorithm described above, and measured the MSE between the computed and the actual coordinates. We then computed the image error in pixels after reprojecting the reconstructed features back to the image. Results are summarized below in Table 4, for 16-18 features points (a selection of corners on the floor, the cars and the calibration box).

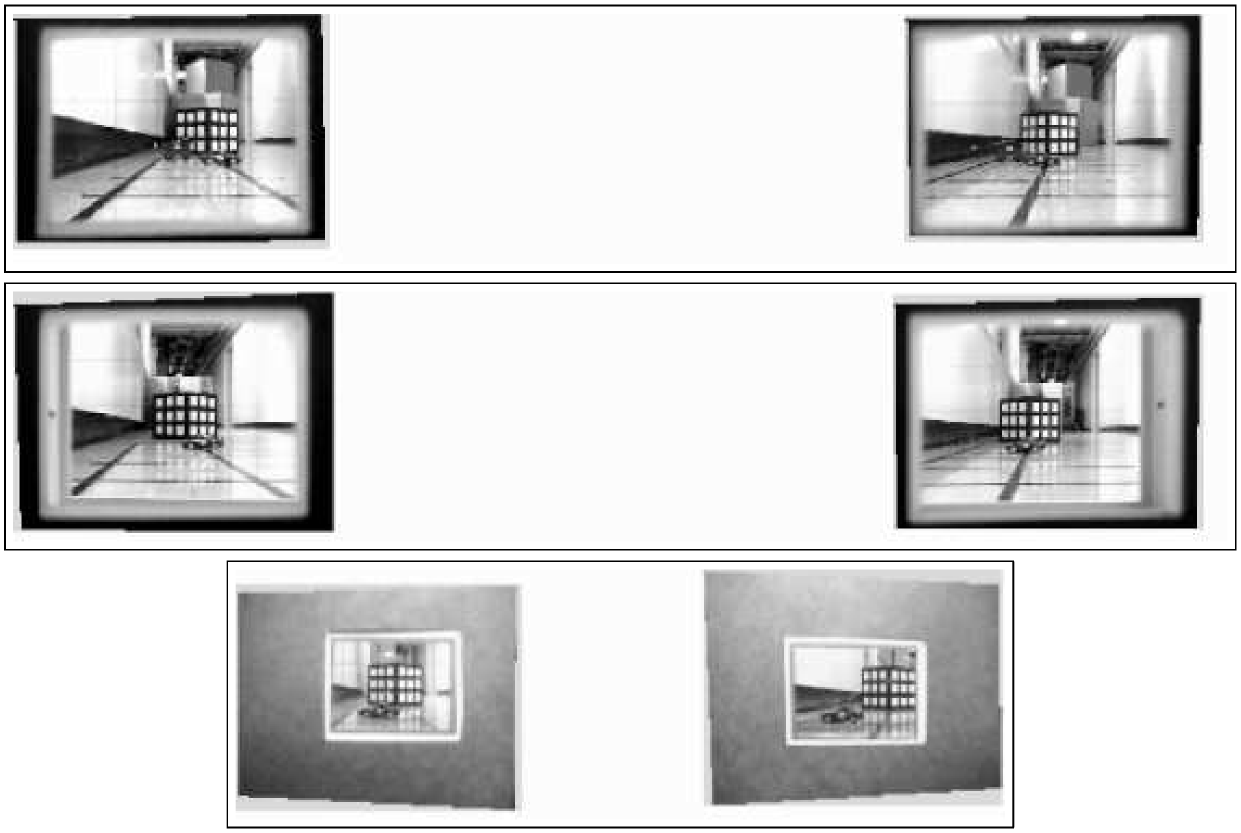


Figure 15: The three image pairs used in the corridor still life experiment, registered with the *reference frame*. The external frame of the *reference frame* is also shown in each case.

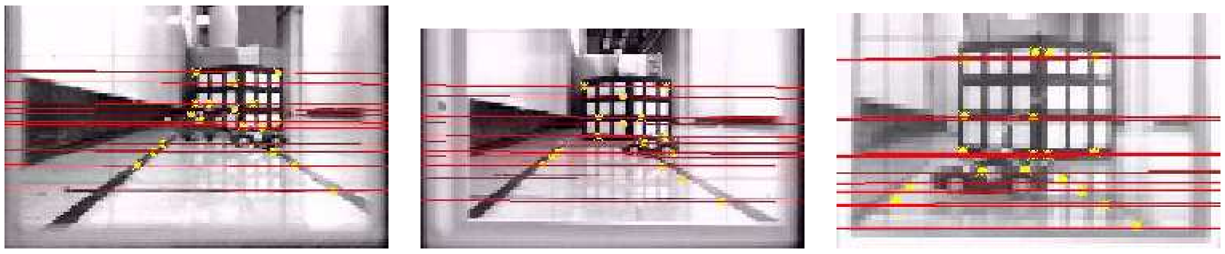


Figure 16: Epipolar geometry results, using our method: the epipolar lines computed for each image point (denoted by a circle+cross), as obtained from the computed fundamental matrix on the right image, are drawn over the left image. It can be seen that all of them pass very close to the matching left image point. Results are shown for the three sequences.

6 Summary and Discussion

We explored the idea where cameras can move around in the world with some ability to see themselves; what the cameras actually see is an aperture rigidly attached to the camera rig, and visible at all times. The registration of the visible apertures with a reference image of the camera's rig eliminates the dependency of image measurements on the camera's internal parameters and external orientation. This greatly simplifies camera calibration, and can be used for tasks that require 3-D metric inference from a few images.

seq. 1 (18 pts)	seq. 2 (16 pts)	seq. 3 (18 pts)
(0.45, 0.1, 1.1)	(0.16, 0.16, 1)	(0.16, 0.16, 2.5)
(1.14, 0.13)	(0.14, 0.12)	(-1.27, 0.1)
(-0.85, 0.11)	(-0.77, 0.13)	(1.13, 0.07)

Table 4: Reconstruction results for the 3 image pairs used in the second experiment: the first line gives the 3-D error, the mean square error in inches (where the distance of the furthest object, the calibration box, is 72 inches) of the 3-D reconstruction in the *3-D reference coordinate system* (error is absolute - no alignment); the second and third lines give image reprojection errors in the left and right image respectively.

Some advantages of this approach (as compared to existing methods) include: *Flexibility* - our cameras may change their parameters from frame to frame in an unconstrained way since the calibration process is on-going and can be done for each pair of images; similarly the scene can change. *Computational simplicity* - our method only involves the solution of linear systems of equations. *Automation and low cost* - the proposed registration can be accomplished automatically and without the supervision of an expert user or the use of expensive equipment.

To study the usefulness of the approach, we studied two applications: gesture recognition and 3-D reconstruction. The results of pointing target detection show that our approach can be accurate enough for practical applications of human-machine interaction, replacing the remote control or laser pointer for such purposes as on-screen selection (e.g., in the context of a computer game). We note that, especially when it concerns very young children, pointing is a much preferred mode of interaction than the alternative (mouse).

Appendix: Geometry on the Reference Plane

This section summarizes related derivations that appeared in [23].

The perspective projection of a point P_i in space to an image plane can be written as $p_{it} = M_t P_i$, where p_{it} and P_i are the point homogeneous coordinates in $2D$ and $3D$ respectively, and M_t is the 3×4 projection matrix describing the camera t. M_t depends on the orientation of the image plane and the location of the camera center P_t. In the following derivation we break down the projection into 2 operations: the projection of the $3D$ world onto a $2D$ reference plane Π through the focal-point P_t, followed by a $2D$ projective transformation (homography) which maps the reference plane Π to the image plane of camera t.

Reference plane coordinate system

Let Π denote a (real or virtual) planar surface in the scene. We call Π the "reference plane". We define a $3D$ Cartesian coordinate system whose $X - Y$ axes span the reference plane Π, and the Z direction is perpendicular to Π. We call this system the "reference plane coordinate system".

An object, composed of n points in $3D$ space, is represented in the reference plane coordinate system by the **shape matrix $\mathbf{P}$**, the following $4 \times n$ matrix:

$$\mathbf{P} = \begin{bmatrix} 0 & a_1 & a_2 & a_3 & a_4 & & X_i & X_{i+1} & \\ 0 & b_1 & b_2 & b_3 & b_4 & \cdots & Y_i & Y_{i+1} & \cdots \\ 1 & 0 & 0 & 0 & 0 & & Z_i & Z_{i+1} & \\ 0 & 1 & 1 & 1 & 1 & & W_i & W_{i+1} & \end{bmatrix} \tag{3}$$

Columns $[a_i \; b_i \; 0 \; 1]$ are the coordinates of points on the reference plane Π, $[0 \; 0 \; 1 \; 0]$ is a point at ∞ on the line through the origin which is perpendicular to the reference plane, and columns $[X_i \; Y_i \; Z_i \; W_i]^T$ are the coordinates of general points P_i on the object.

We note that every object can be represented in this way, but the representation is *not* unique: a transformation from another projective coordinate system to this particular one is determined only up to Z

axis scaling. This ambiguity comes about because the standard projective basis requires that no four points are co-planar, whereas the basis of our system includes a plane.

Projection on the reference plane

The $3D$ scene points are first projected onto the reference plane Π through the focal-point P_t of the camera. This forms a "virtual" image on Π. We refer to this image as the "reference-plane" image.

For the object-shape matrix P defined in (3) and camera t, we define the following matrix $\mathbf{p_t}$ of reference-image points:

$$\mathbf{p_t} = \begin{bmatrix} \alpha_t & a_1 & a_2 & a_3 & a_4 & & x_{it} & x_{(i+1)t} & \\ \beta_t & b_1 & b_2 & b_3 & b_4 & \cdots & y_{it} & y_{(i+1)t} & \cdots \\ \gamma_t & 1 & 1 & 1 & 1 & & w_{it} & w_{(i+1)t} & \end{bmatrix} \tag{4}$$

Note that the first two coordinates of points $1, \ldots 4$, which are physically located on the reference plane Π, are the same as the corresponding coordinates in their $3D$ representation. This is a direct consequence of our choice of coordinate system for $3D$ representation.

Next, it can be shown (or, more easily, verified) that the projection matrix M_t from (3) to (4) is:

$$M_t = \begin{bmatrix} \delta_t & 0 & \alpha_t & 0 \\ 0 & \delta_t & \beta_t & 0 \\ 0 & 0 & \gamma_t & \delta_t \end{bmatrix}$$

We impose the constraint that the focal point of the camera cannot be possibly seen in the (reference-plane) image; thus it is the null vector of M_t. We denote the $3D$ coordinates of the focal point P_t of the camera t by $[X_t\ Y_t\ Z_t\ W_t]$, then

$$\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \propto \begin{bmatrix} \delta_t & 0 & \alpha_t & 0 \\ 0 & \delta_t & \beta_t & 0 \\ 0 & 0 & \gamma_t & \delta_t \end{bmatrix} \cdot \begin{bmatrix} X_t \\ Y_t \\ Z_t \\ W_t \end{bmatrix} \quad \Longrightarrow \begin{cases} 0 = \delta_t X_t + \alpha_t Z_t \\ 0 = \delta_t Y_t + \beta_t Z_t \\ 0 = \delta_t W_t + \gamma_t Z_t \end{cases}$$

and the solution is:

$$[\delta_t\ \alpha_t\ \beta_t\ \gamma_t] \propto [-Z_t\ X_t\ Y_t\ W_t]$$

The normalized reference-plane image $\mathbf{p_t}$ is therefore obtained by the following projection:

$$\mathbf{p_t} \propto \begin{bmatrix} -Z_t & 0 & X_t & 0 \\ 0 & -Z_t & Y_t & 0 \\ 0 & 0 & W_t & -Z_t \end{bmatrix} \mathbf{P} \tag{5}$$

Epipolar Geometry on the Reference Plane:

Let $p_{it} = [x_{it}\ y_{it}\ w_{it}]^T$ denote the reference-plane image position of the object point P_i projected through the camera focal-point P_t (onto Π). From (5) we have that:

$$p_{it} = \begin{bmatrix} x_{it} \\ y_{it} \\ w_{it} \end{bmatrix} \propto \begin{bmatrix} -Z_t & 0 & X_t & 0 \\ 0 & -Z_t & Y_t & 0 \\ 0 & 0 & W_t & -Z_t \end{bmatrix} \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ W_i \end{bmatrix} \tag{6}$$

Observe that each point on the reference-plane image imposes 2 constraints relating to the coordinates of the object point P_i and the camera center P_t, which follow immediately from (6):

$$\frac{x_{it}}{w_{it}} = \frac{-Z_t X_i + X_t Z_i}{-Z_t W_i + W_t Z_i}, \qquad \frac{y_{it}}{w_{it}} = \frac{-Z_t Y_i + Y_t Z_i}{-Z_t W_i + W_t Z_i} \tag{7}$$

The epipolar geometry is now obtained by the elimination of the $3D$ point coordinates from (7). In the following derivation we followed the method described in [6]. First, we rewrite (7) and note that each camera t imposes 2 constraints on the $3D$ point coordinates

$$\begin{pmatrix} w_{it}Z_t & 0 & (x_{it}W_t - w_{it}X_t) & -x_{it}Z_t \\ 0 & w_{it}Z_t & (y_{it}W_t - w_{it}Y_t) & -y_{it}Z_t \end{pmatrix} \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ W_i \end{bmatrix} = 0$$

Two cameras t and t' provide four constraints. Since they have a non-trivial solution $[X_i\ Y_i\ Z_i\ W_i]$, the determinant of their matrix must equal 0. This yields a relation between the two focal points $P_t = [X_t\ Y_t\ Z_t\ W_t]$ and $P_t' = [X_t'\ Y_t'\ Z_t'\ W_t']$, and the image points p_{it} and p_{it}' of the point P_i in the two corresponding reference-plane images. This relation can be rewritten as:

$$p_{it}^T\ F\ p_{it}' = 0, \quad F_{ts} = \begin{bmatrix} 0 & Z_t'W_t - Z_tW_t' & -Z_t'Y_t + Z_tY_t' \\ -Z_t'W_t + Z_tW_t' & 0 & Z_t'X_t - Z_tX_t' \\ Z_t'Y_t - Z_tY_t' & -Z_t'X_t + Z_tX_t' & 0 \end{bmatrix} \tag{8}$$

and F is the "fundamental" matrix. Eq. (2) follows from the above by scaling F by $\frac{1}{Z_t'Z_t}$.

Acknowledgements:

The use of flicker light for detection was suggested to us by Misha Pavel. We owe many thanks to Tom Marshal who built the device for us, and to Mircea Nicolesu who designed and implemented the tracker.

References

[1] M. Armstrong, A. Zisserman, and R. Hartley. Euclidean reconstruction from image triplets. In *Proceedings of the 4th European Conference on Computer Vision*, pages 3–16, Cambridge, UK, April 1996.

[2] S. Bougnoux. From projective to Euclidean space under any practical situation, a criticism of self-calibration. In *Proceedings of the 6th International Conference on Computer Vision*, pages 790–796, Bombay, 1998. IEEE, Washington, DC.

[3] R. Cipolla and N. J. Hollinghurst, "Human–robot interface by pointing with uncalibrated stereo vision," in *Image and Vision Computing*, 14(3):171–178, 1996.

[4] A. J. Colmenarez, B. Frey, and T. S. Huang, "Detection and Tracking of Faces and Facial Features," in *Proc. ICIP*, 1998.

[5] O. Faugeras. *Three-Dimensional Computer Vision.* MIT Press, Cambridge, MA, 1993.

[6] O. Faugeras and B. Mourrain. On the geometry and algebra of the point and line correspondences between N images. In *Proc. ECW.* Xidian University Press, 1995.

[7] M. Fukumoto, Y. Suenaga, and K. Mase, "Finger-pointer: pointing interface by image processing," Comput. & Graphics 18(5):633-642, 1994.

[8] C. Harris and M.J. Stephens. A combined corner and edge detector. *Alvey88*, pages 147–152, 1988.

[9] R. Hartley. Euclidean reconstruction from uncalibrated views. In J.L. Mundy, A. Zisserman, and D. Forsyth, editors, *Applications of invariance in computer vision*, volume 825 of *Lecture Notes In Computer Science*, pages 237–256. Springer Verlag, 1994.

[10] R. Hartley. In defence of the 8-point algorithm. In *Proceedings of the 5th International Conference on Computer Vision*, pages 1064–1070, Cambridge, MA, 1995. IEEE, Washington, DC.

[11] A. Heyden and K. Aström. Euclidean reconstruction from constant intrinsic parameters. In *Proceedings Int. Conf. on Pattern Recognition*, pages 339–343. IEEE Computer Soc. Press, 1996.

[12] N. Jojic, B. Brumitt, B. Meyers, Steve Harris, and Thomas Huang, "Detection and estimation of pointing gestures in real-time stereo sequences," in *Proc. ICAFGR*, 2000.

[13] R. E. Kahn, M. J. Swain, P. N. Prokopiwicz and J. Firby, "Gesture recognition using the Perseus architecture," in *Proc. CVPR*, 1996.

[14] D. Kortenkamp, E. Huber, and P. R. Bonasso, "Recognizing and Interpreting Gestures within the Context of an Intelligent Robot Control Architecture," Working Notes: 1995 AAAI Fall Symposium on Embodied Language and Action, 1995.

[15] R. Kumar, P. Anandan, and K. Hanna. Direct recovery of shape from multiple views: a parallax based approach. In *Proceedings Int. Conf. on Pattern Recognition*, 1994.

[16] R. Koch M. Pollefeys and L. VanGool. Self-calibration and metric reconstruction inspite of varying and unknown intrinsic camera parameters. *International Journal of Computer Vision*, 32(1):7–25, 1999.

[17] S. J. Maybank and O. D. Faugeras. A theory of self-calibration of a moving camera. *International Journal of Computer Vision*, 8(2):123–152, 1992.

[18] C. Nölker and H. Ritter, "GREFIT: Visual Recognition of Hand Postures," in A. Braffort et al. (Eds.): GW'99, LNAI 1739, pp. 61–73, 1999.

[19] V. I. Pavlovic, R. Sharma, and Thomas Huang, "Gestural interface to a visual computing environment for molecular biologist," in *Proc. ICAFGR*, 1996.

[20] A. Shashua and N. Navab. Relative affine structure: Theory and application to 3D reconstruction from perspective views. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition*, pages 483–489, 1994.

[21] G. Stein. Accurate internal camera calibration using rotation, with analysis of sources of errors. In *Proceedings of the 5th International Conference on Computer Vision*, pages 230–236, Cambridge, MA, 1995. IEEE, Washington, DC.

[22] T. Darrell, "A radial cumulative transform for robust image correspondence," in *Proc. CVPR*, 1998.

[23] D. Weinshall, P. Anandan, and M. Irani. From ordinal to Euclidean reconstruction with partial scene calibration. In R. Koch and L. VanGool, editors, *Proceedings of SMILE Workshop on Structure from Multiple Images in Large Scale Environments*, Lecture Notes in Computer Science. Springer Verlag, June 1998.

[24] A. Wu, M. Shah, N. da Vitoria Lobo, "A virtual 3D blackboard: 3D finger detection using a single camera," in *Proc. ICAFGR*, 2000.

[25] O. Faugeras Z. Zhang and R. Deriche. An effective technique for calibrating a binocular stereo through projective reconstruction using both a calibration object and the environment. *Videre: Journal of Computer Vision Research*, 1(1):58–68, 1997.

[26] Z. Zhang. A flexible new technique for camera calibration. Technical report, msr-tr-98-71, Microsoft Research, Mircrosoft Corporation, One Microsoft Way, Redmond, WA 98052, 1998.