Augmented Reality

Integrating Computer Graphics with Computer Vision

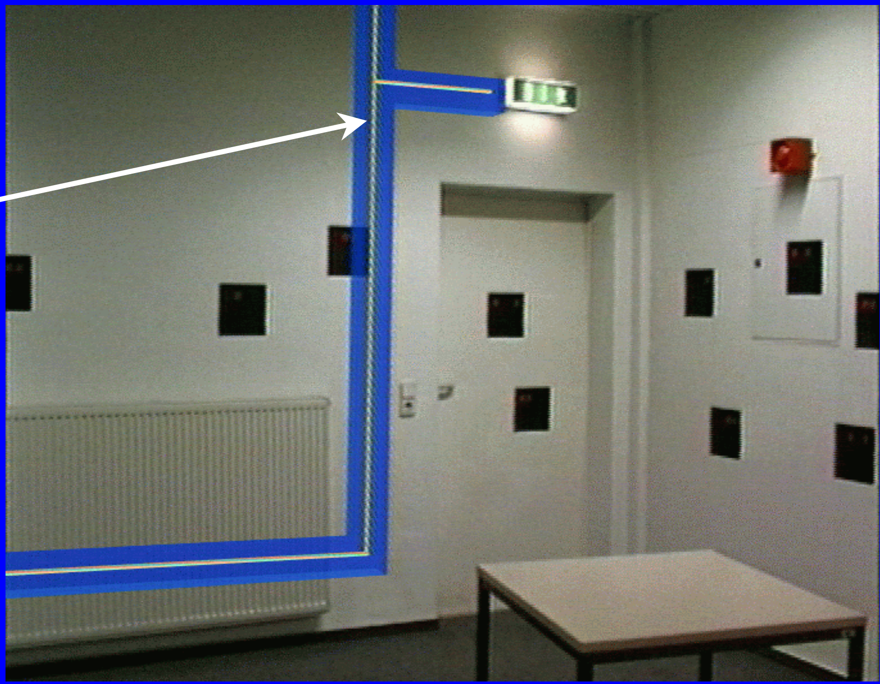
Mihran Tuceryan

Definition

- Combines real and virtual worlds and objects
- It is interactive and real-time
- The interaction and alignment of real and virtual objects happens in 3D (I.e., not 2D overlays).

An Example Application

Electric wires shown as “augmentation” in a physical room.



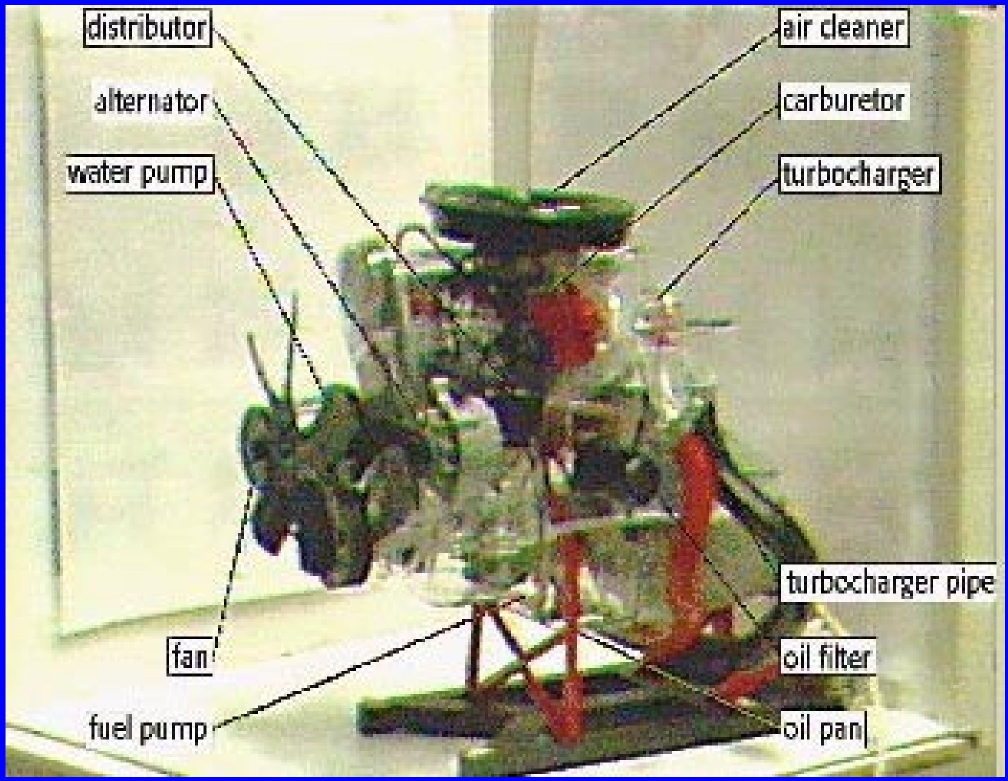
Motivation

Claim:

Augmented Reality can be deployed in much more useful applications in the real world than virtual reality.

Possible Applications

Maintenance and Repair



Ref: ECRC

Possible Applications

- Architecture and Construction

Before

After

Images: courtesy of CICC project at ZGDV, Fraunhofer IGD

Possible Applications

- Interior Design

Ref: ECRC

Possible Applications

- Aid in construction sites: X-ray view of a wall

Image: courtesy of ZGDV/Fraunhofer IGD and Holzmann AG

Possible Applications

- Aid for assembly in car manufacturing: instructions for two-handed screw fixing

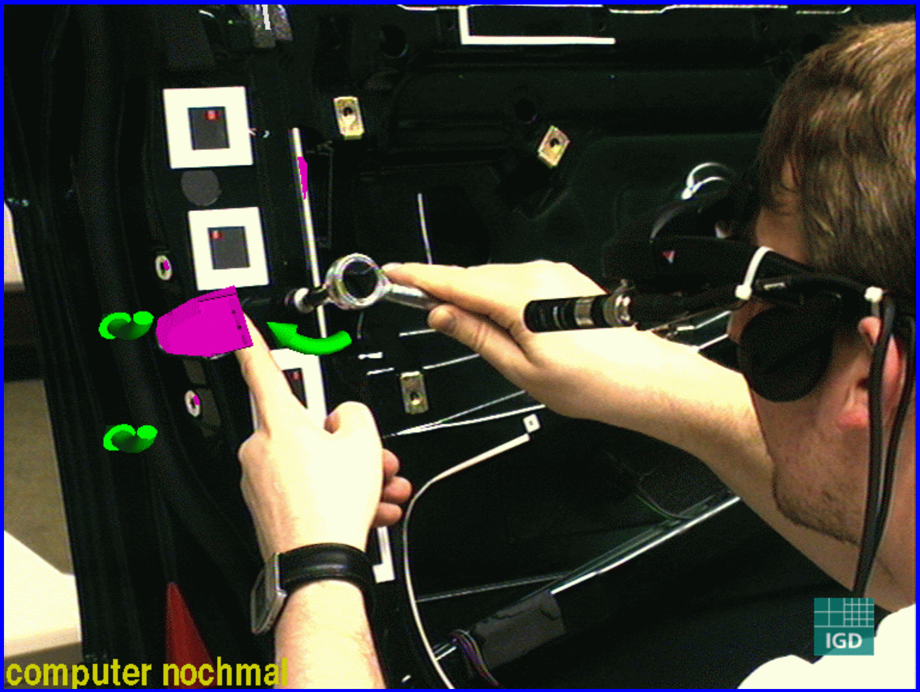


Image: courtesy of ZGDV/Fraunhofer IGD, data courtesy of BMW

Possible Applications

- Aid for assembly in car manufacturing: lever inside the door to be pushed to right position

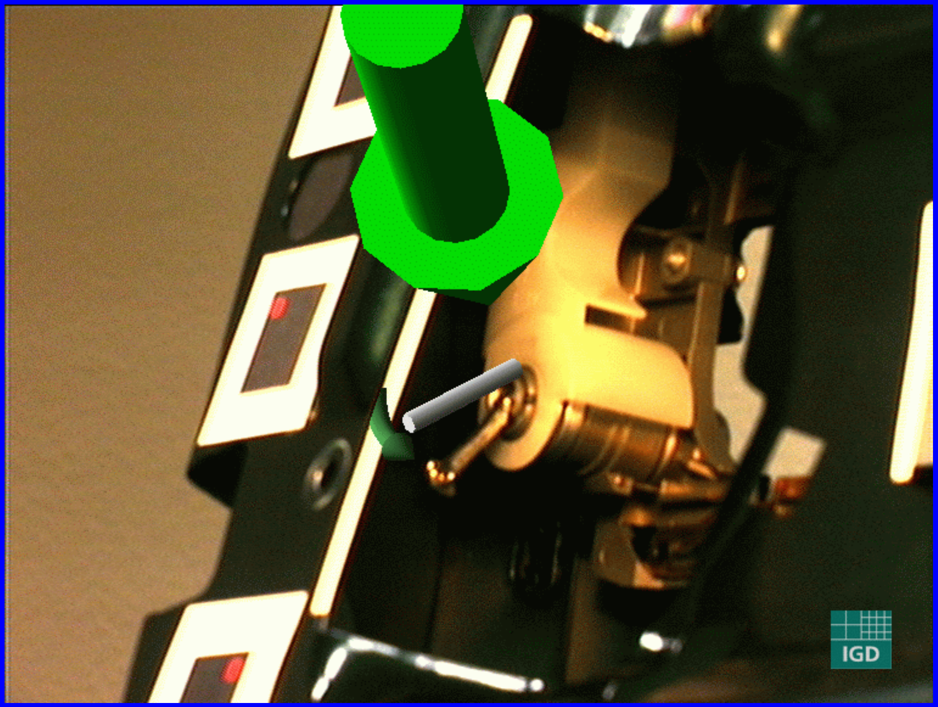


Image: courtesy of ZGDV/Fraunhofer IGD, data courtesy of BMW

More Possible Applications

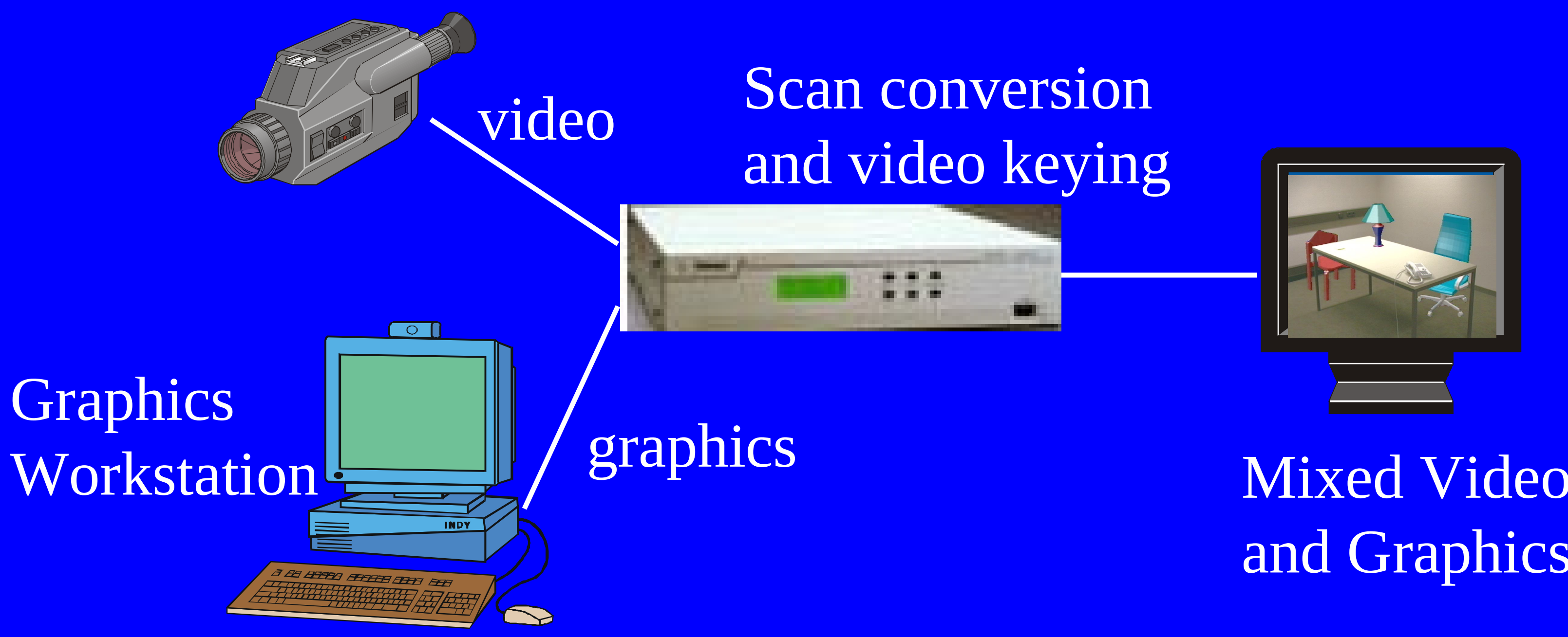
- Computer Aided Surgery
- Fashion Design or Apparel Retail
- Circuit Board Diagnostics and Repair
- Facilities Maintenance
- Industrial Plant Maintenance
- Road Repair and Maintenance

Major Components

- Display subsystem
- Rendering subsystem
- Video input and image understanding subsystem
- Interaction subsystem
- Registration and tracking subsystem

Display Technologies

See-through video display technology



Display Technologies

- See-through Head Mounted Displays (see-through HMD)
- World seen directly by the eye with graphics superimposed

Graphics
Workstation

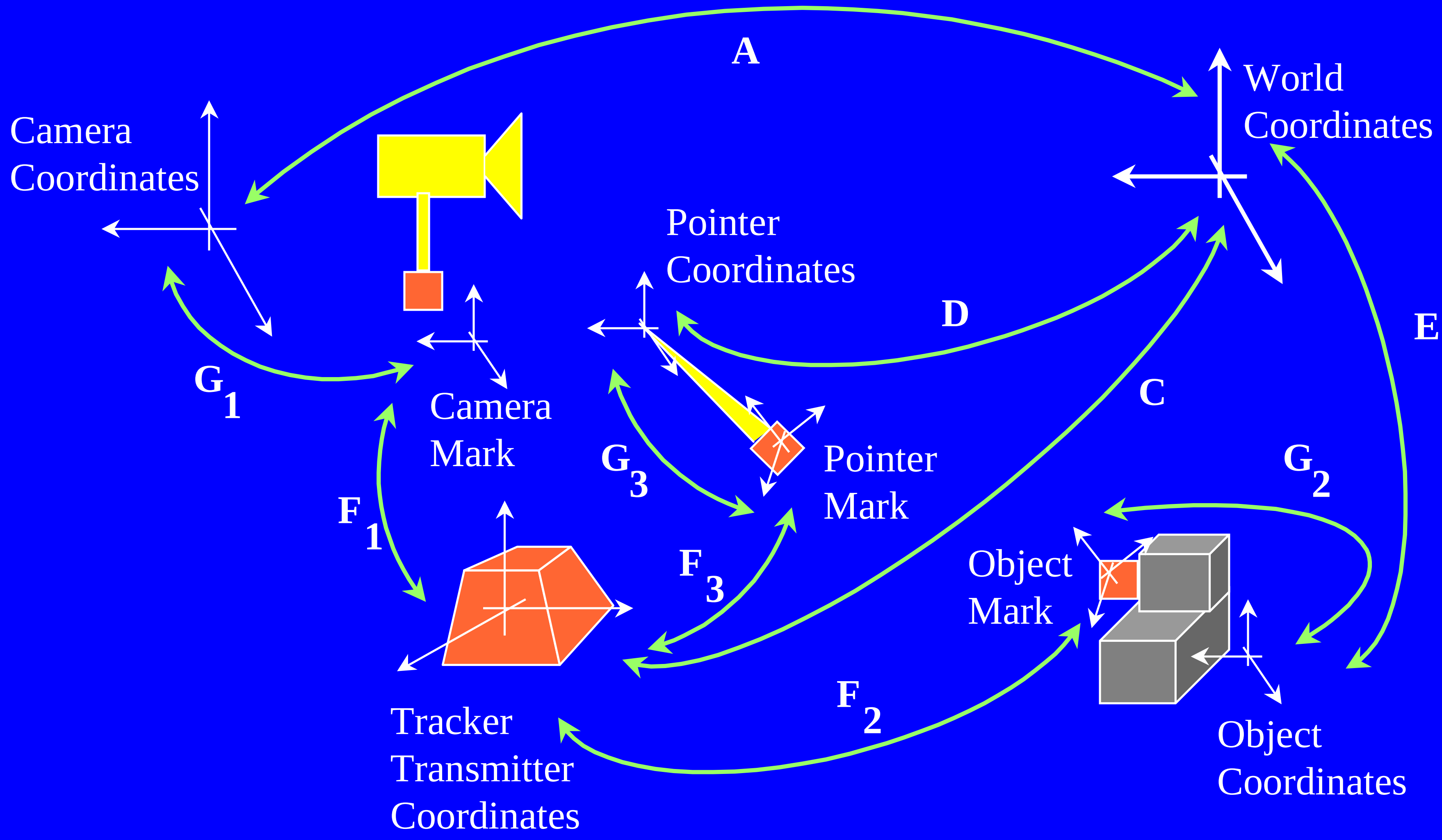
Rendering subsystem

- VR technology
  - 3D real-time rendering
  - 3D interaction
  - head and camera tracking

Virtual Camera

- 3D graphics rendered through a virtual camera
- Realistic augmentation (with registration and correct optics)
  - ==> virtual camera matches real camera or optics of human eye.

Coordinate Systems in see-through video



Calibration

- All the coordinate transforms in the previous slide need to be estimated.
- This is done through calibration.

Image Calibration

- Scan converter can access an arbitrary region of the screen to be mixed with graphics.
- Modeled as 2D translation and scaling

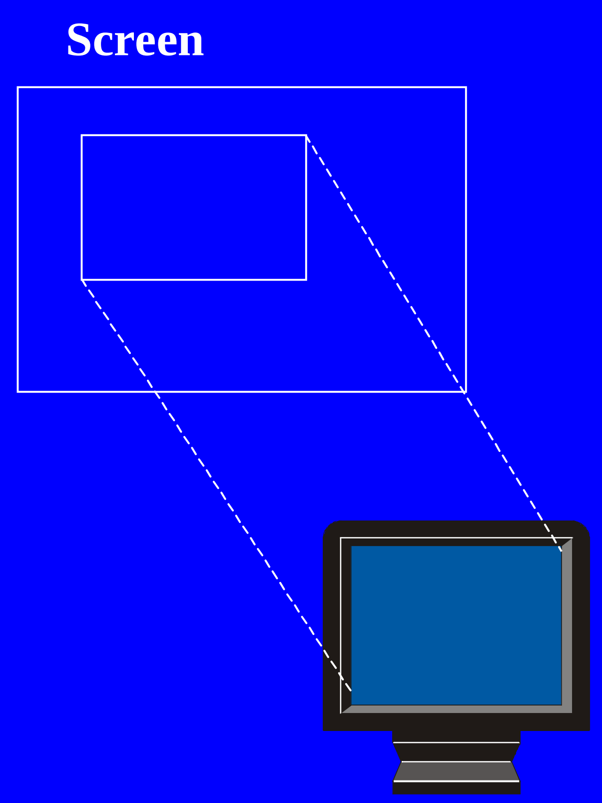
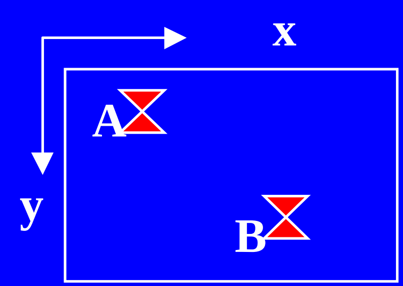
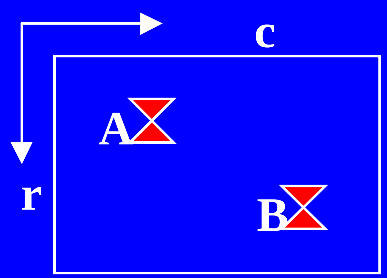


Image Calibration

Image of two known points displayed and the result is captured



Computer Generated image



Grabbed image

Calibration estimates k_x, k_y, t_x, t_y by equations

$$c = k_x x + t_x$$

$$r = k_y y + t_y$$

$$\tilde{k}_x = (c_B - c_A)/(x_B - x_A)$$

$$\tilde{k}_y = (r_B - r_A)/(y_B - y_A)$$

$$t_x = c_A - \tilde{k}_x x_A$$

$$t_y = r_A - \tilde{k}_y y_A$$

Camera Calibration

- Estimates the initial pose of the camera:
  R: rotation matrix
  T: translation
- Estimates the camera intrinsic parameters:
  f: focal length
  (r_0, c_0): image center
  (s_x, s_y): scale factors

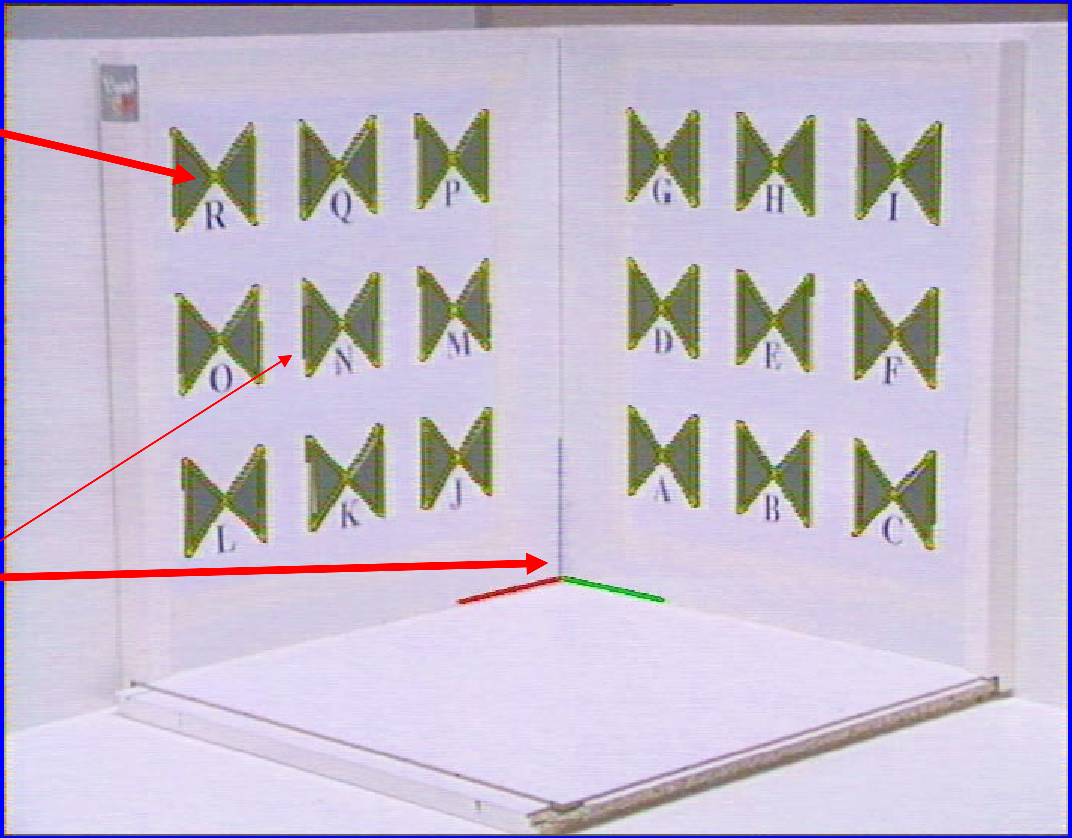
Camera Calibration (continued)

- Many camera calibration methods exist in computer vision literature:
  - R. Tsai
  - Weng, Cohen, Herniou
  - …
- The estimated camera parameters are used in the virtual camera with which 3D graphics are rendered.

Camera Calibration (continued)

Points are picked from calibration jig whose 3D coordinates are known

The model of calibration jig is rendered using a virtual camera with the estimated parameters. The rendered graphics is superimposed on the image of calibration jig.



Camera Calibration (continued)

(r_0, c_0) : image center

(s_u, s_v) : scale factors in x and y (or u and v) directions

f : focal length

$\mathbf{R} = [r_{ij}]$: rotation matrix for the camera

$\mathbf{T} = \begin{bmatrix} t_1 \\ t_2 \\ t_3 \end{bmatrix}$: translation vector for camera

Camera Calibration (continued)

Camera Equations given by:

$$\frac{r - r_0}{s_u f} = \frac{r - r_0}{f_u} = \frac{r_{11}x + r_{12}y + r_{13}z + t_1}{r_{31}x + r_{32}y + r_{33}z + t_3}$$

$$\frac{c - c_0}{s_v f} = \frac{c - c_0}{f_v} = \frac{r_{21}x + r_{22}y + r_{23}z + t_2}{r_{31}x + r_{32}y + r_{33}z + t_3}$$

$(r, c):$ image coordinates in rows and columns

Camera Calibration (continued)

- **Collect image data points (r_i, c_i) corresponding to known 3D calibration points (x_i,y_i,z_i)**
- **These coordinates are corrected using the result of image calibration.**
- **Substitute into camera equations in previous slide ==> gives 2 equations per calibration point.**

Camera Calibration (continued)

- Variable substitution to linearize camera equations:

$$\left.\begin{aligned} \mathbf{W_1} &= f_u \mathbf{R}_1 + r_0 \mathbf{R_3} \\ \mathbf{W_2} &= f_v \mathbf{R_2} + c_0 \mathbf{R_3} \\ \mathbf{W_3} &= \mathbf{R_3} \end{aligned}\right\} \textbf{ where } \left\{\begin{aligned} \mathbf{R} &= \begin{bmatrix} \mathbf{R}_\mathbf{1}^T \\ \mathbf{R}_2^T \\ \mathbf{R}_3^T \end{bmatrix} \\ \mathbf{W}_i &= \begin{bmatrix} w_{i1} \\ w_{i2} \\ w_{i3} \end{bmatrix} \end{aligned}\right.$$

$$w_4 = f_u t_1 + r_0 t_3$$

$$w_5 = f_v t_2 + c_0 t_3$$

$$w_6 = t_3$$

Camera Calibration (continued)

Solve the homogeneous equation

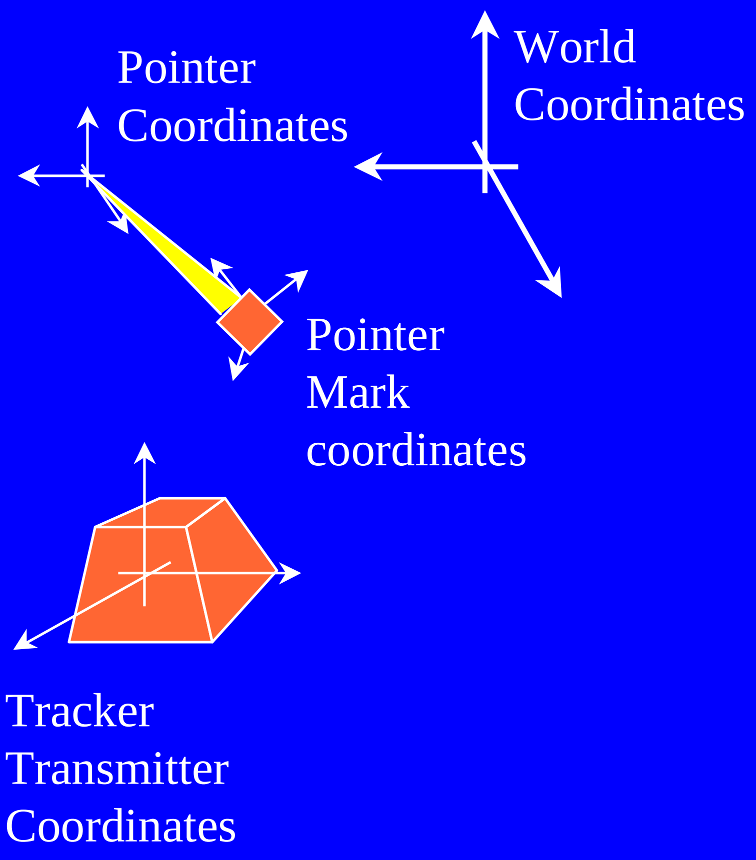
$$\mathbf{AW} = \mathbf{0}$$

where

$$A = \begin{bmatrix} -x_1 & -y_1 & -z_1 & 0 & 0 & 0 & r_1 x_1 & r_1 y_1 & r_1 z_1 & -1 & 0 & r_1 \\ 0 & 0 & 0 & -x_1 & -y_1 & -z_1 & c_1 x_1 & c_1 y_1 & c_1 z_1 & 0 & -1 & c_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ -x_n & -y_n & -z_n & 0 & 0 & 0 & r_n x_n & r_n y_n & r_n z_n & -1 & 0 & r_n \\ 0 & 0 & 0 & -x_n & -y_n & -z_n & c_n x_n & c_n y_n & c_n z_n & 0 & -1 & c_n \end{bmatrix}$$

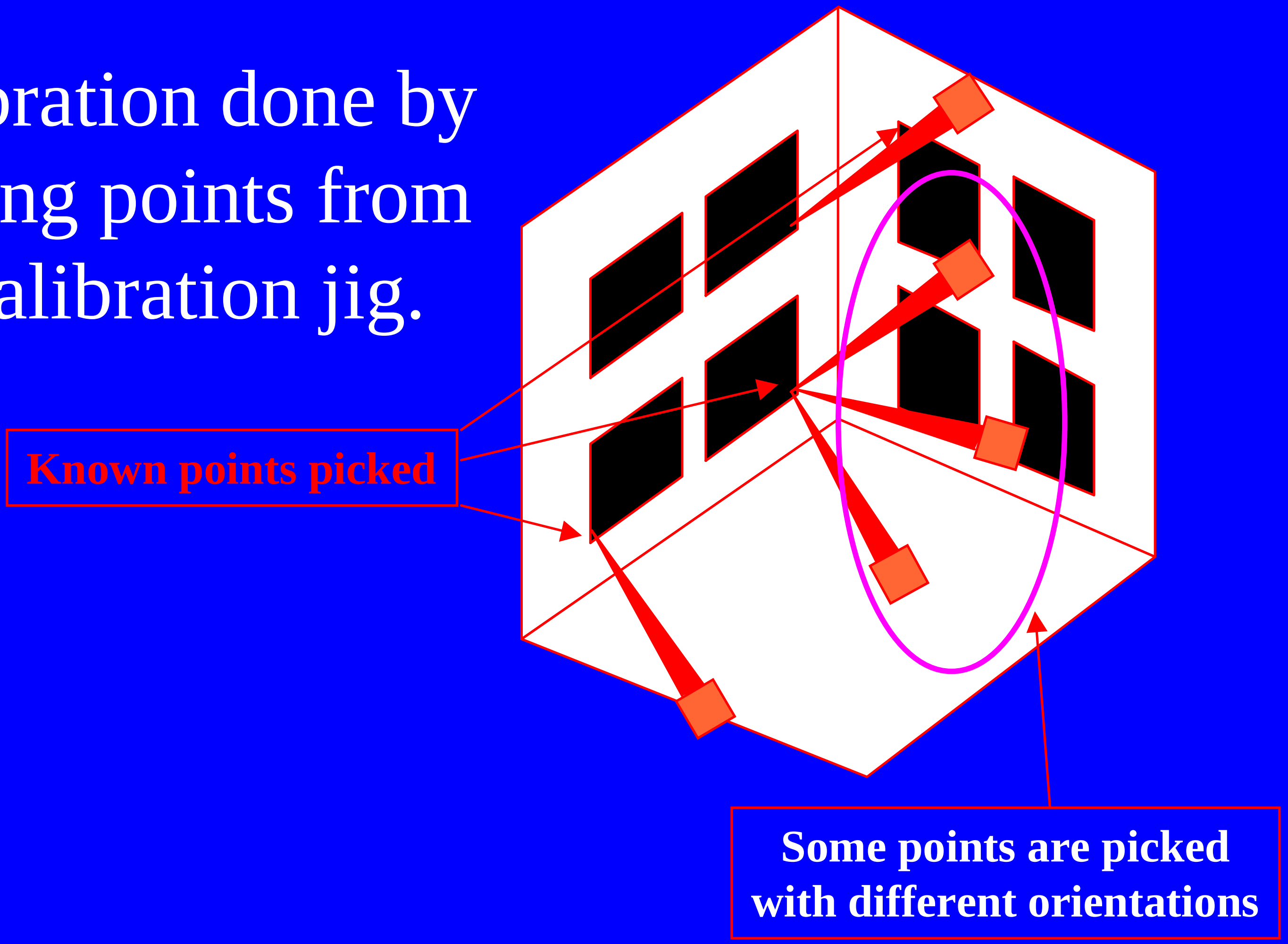
Pointer calibration

- 3D pointer is implemented with magnetic tracker.
- The tracker coordinate system and the pointer tip needs to be related to the world coordinate system.
- This is done through a pointer calibration process.



Tracker and pointer calibration

- Calibration done by picking points from the calibration jig.



Pointer Calibration

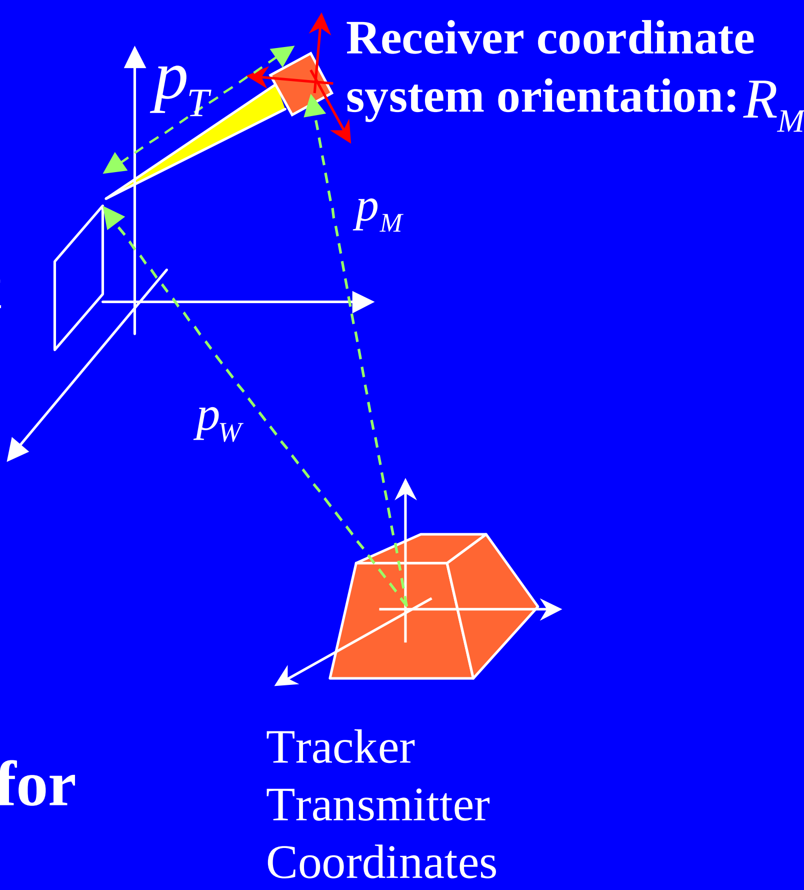
The tip of the pointer is given by the equation:

$$p_W = p_M + R_M \, p_T$$

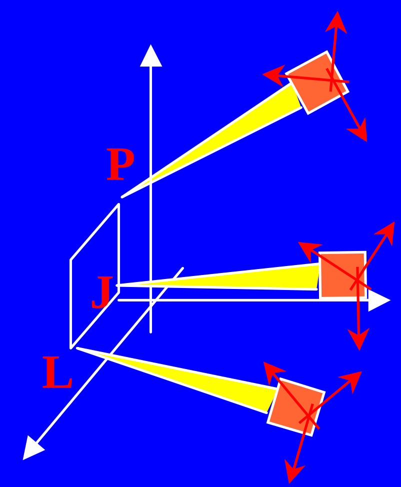
Reading the pointer at n different orientations, we get the equation

$$\begin{pmatrix} I & -R_{M_1} \\ I & -R_{M_2} \\ \vdots & \vdots \\ I & -R_{M_n} \end{pmatrix} \begin{pmatrix} p_W \\ p_T \end{pmatrix} = \begin{pmatrix} p_{M_1} \\ p_{M_2} \\ \vdots \\ p_{M_n} \end{pmatrix}$$

Solve the overdetermined system for p_T and p_W



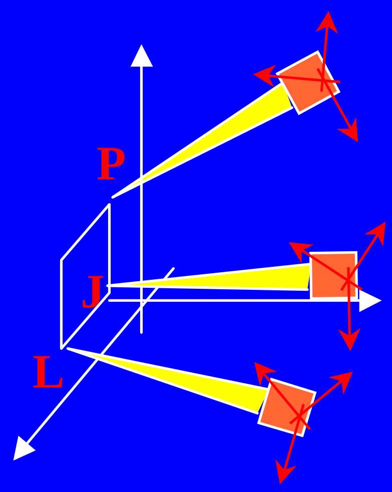
Tracker transmitter calibration



- Estimate position of transmitter in world coordinates
- Read three known points on calibration jig.
- The origin of the world, p_0, with respect to the transmitter is solved from $p_{w_J} = p_{M_J} + R_{M_J} p_0$ after the rotation R_{M_J} is estimated

Tracker Transmitter Calibration

- The rotation matrix R_{M_J} is estimated by first computing the axes:



$$\mathbf{x} = p_{W_L} - p_{W_J}$$

$$\mathbf{z} = p_{W_P} - p_{W_J}$$

$$\mathbf{y} = \frac{\mathbf{z}}{\|\mathbf{z}\|} \times \frac{\mathbf{x}}{\|\mathbf{x}\|}$$

$$R_{M_J} = [\frac{\mathbf{x}}{\|\mathbf{x}\|}\ \mathbf{y}\ \frac{\mathbf{z}}{\|\mathbf{z}\|}]^T$$

Registration

- Aligning 3D models of objects with their real counterparts.
- Alignment errors must be extremely small.

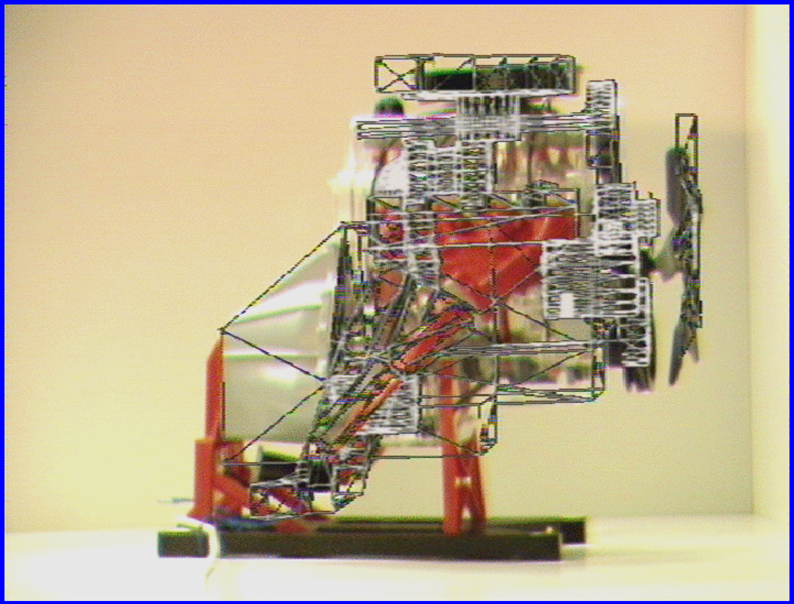
Registration (examples)

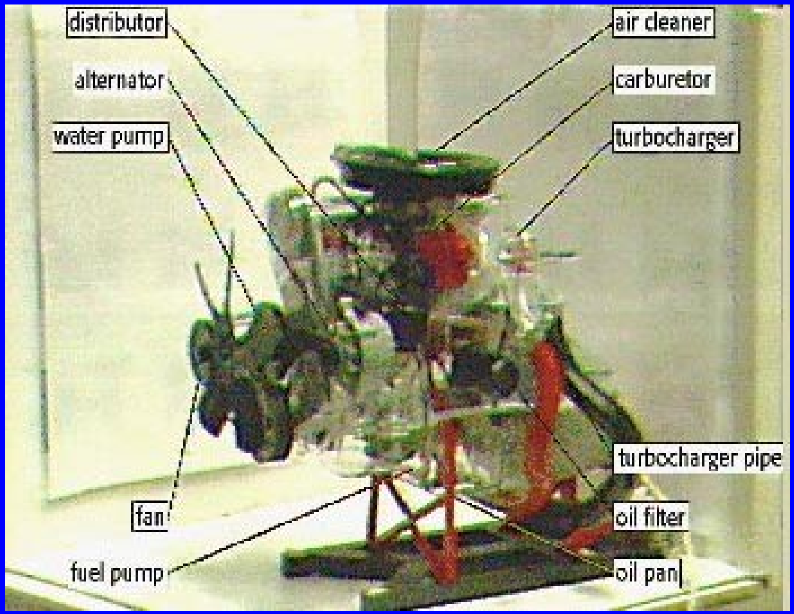
Example applications where registration is used

A 3D model of the table is registered with the physical table to...

…obtain the effect of occlusion by using the geometry of the 3D model

Registration (Examples)





A 3D model of the engine is registered with the physical engine to...

… label the parts of the engine using the parts knowledge encoded in the 3D model.

Registration Methods

- Image based landmarks for computing object pose.
- 3D pointer based for computing object pose.
- Automatic registration of objects in 2D images using computer vision techniques.

Image Based Landmarks

- Camera parameters known from camera calibration.
- 3D coordinates of landmarks known from object model.
- 2D image coordinates of landmark points picked manually.

Image Based Landmarks

- Plug the known values into the camera equations:

$$(r_i - r_0)x_i r_{31} + (r_i - r_0)y_i r_{32} + (r_i - r_0)z_i r_{33}$$
$$+ (r_i - r_0)t_3 - f_u x_i r_{11} - f_u y_i r_{12} - f_u z_i r_{13} - f_u t_1 = 0$$
$$(c_i - c_0)x_i r_{31} + (c_i - c_0)y_i r_{32} + (c_i - c_0)z_i r_{33}$$
$$+ (c_i - c_0)t_3 - f_v x_i r_{21} - f_v y_i r_{22} - f_v z_i r_{23} - f_v t_2 = 0$$

- Solve for translation and rotation under the constraint of orthonormal rotation matrix.

Image Based Landmarks

- Solution obtained by minimizing

$$\left\|\mathbf{Ax}\right\|^2 + \alpha\left\|\mathbf{R}^T\mathbf{R} - \mathbf{I}\right\|^2$$

where **x** is the vector of unknowns.

3D Pointer Based Landmarks

- Calibrated 3D pointer device,
- 3D coordinates of landmark points known from the object model,
- 3D landmark points picked on the physical object with the pointer.

3D Pointer Based Landmarks

- The picked 3D landmark points and 3D model landmarks are related to each other with a rigid transformation:

$$p_i^W = \mathbf{R} p_i^L + T \quad \text{for } i = 1, \cdots, n$$

where

p_i^W : world coordinates

p_i^L : landmark coordinates

3D Pointer Based Landmarks

- Solution by minimizing equation:

$$\left\| p_i^W - \mathbf{R} p_i^L - \mathbf{T} \right\|^2 + \alpha \left\| \mathbf{R}^T \mathbf{R} - \mathbf{T} \right\|^2$$

Automatic Registration

- Computer Vision techniques for registering objects with their images and computing object pose
- Features detected automatically and matched to model
- Less general
- Less robust

Registration Pitfalls

- **R** matrix in the solution to these equations may not be a valid rotation matrix.
- One possible approach is to formulate the equations representing the rotation as a quaternion instead of a 3x3 matrix.

Tracking subsystem

- Many tracking system possibilities
  - Magnetic trackers
  - Mechanical trackers
  - Optical trackers
  - Ultrasound trackers
  - Vision based trackers

Magnetic trackers

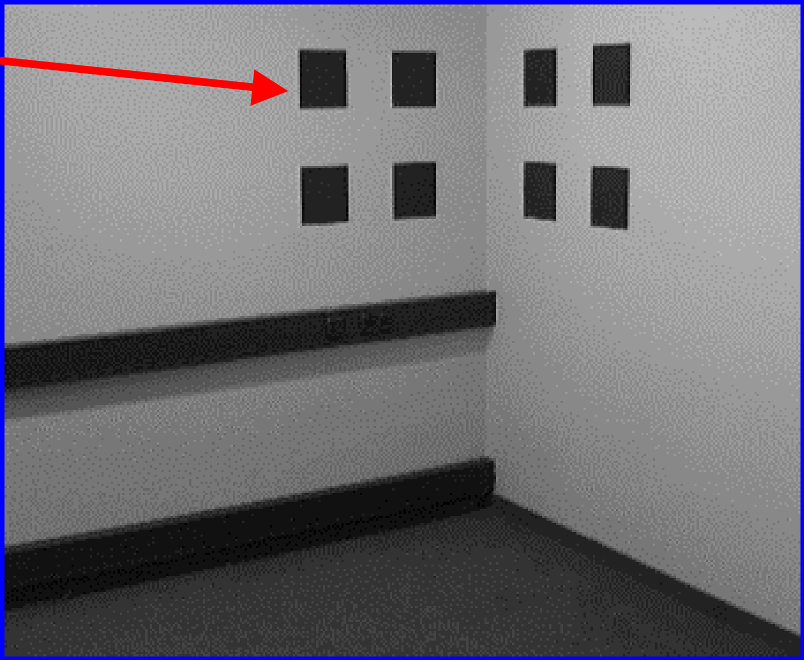
- Objects are tracked by attaching a receiver of magnetic trackers onto the objects.
- The data read from the magnetic tracker is used to update the object-to-world rigid transformation.
- This requires a calibration procedure (as part of object registration) by which the object-mark-to-object transformation is initially estimated.

Vision Based Tracking

- Camera tracking
- Object tracking
  - One implementation using landmarks put in the environment. Camera motion is computed automatically.

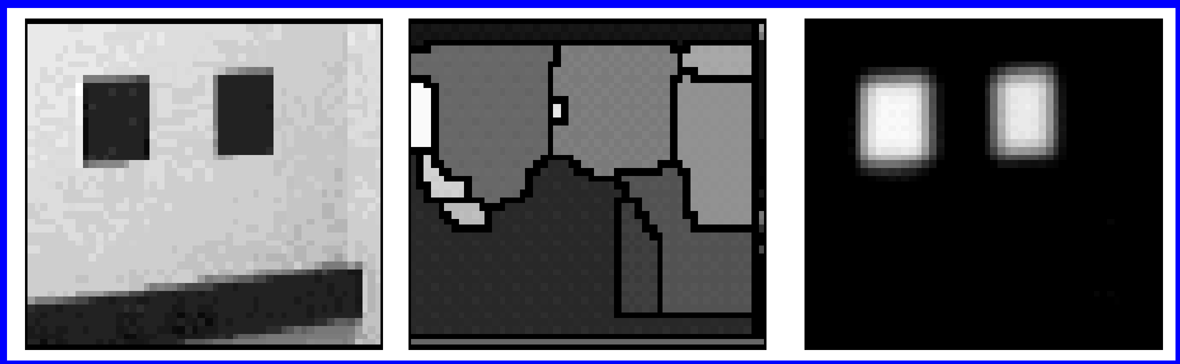
Vision Based Camera Tracking

- Environment modified with landmarks whose 3D coordinates are known
- Landmarks detected automatically
- camera pose extracted from corner coordinates of square landmarks similar to camera calibration procedure.



Vision Based Camera Tracking

- Landmark squares in the environment are detected (segmented) using a watershed transformation algorithm



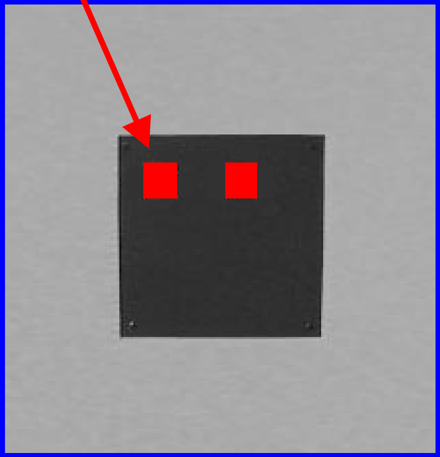
Original image **Watershed transformation** **Results of inside operation for regions of watershed transformation**

Vision Based Camera Tracking

- Landmark squares contain red squares at known positions which encode the identity of the model squares

Red squares are barely visible in the green and blue channels.

Black squares segmented in the green channel.
Red channel is sampled at locations where red dots should be w.r.t. the landmark boundary giving the id of the square.



Vision Based Camera Tracking

Squares detected and room scene augmented with a wall rendered with proper perspective

Vision Based Camera Tracking

- Motion model assumed:

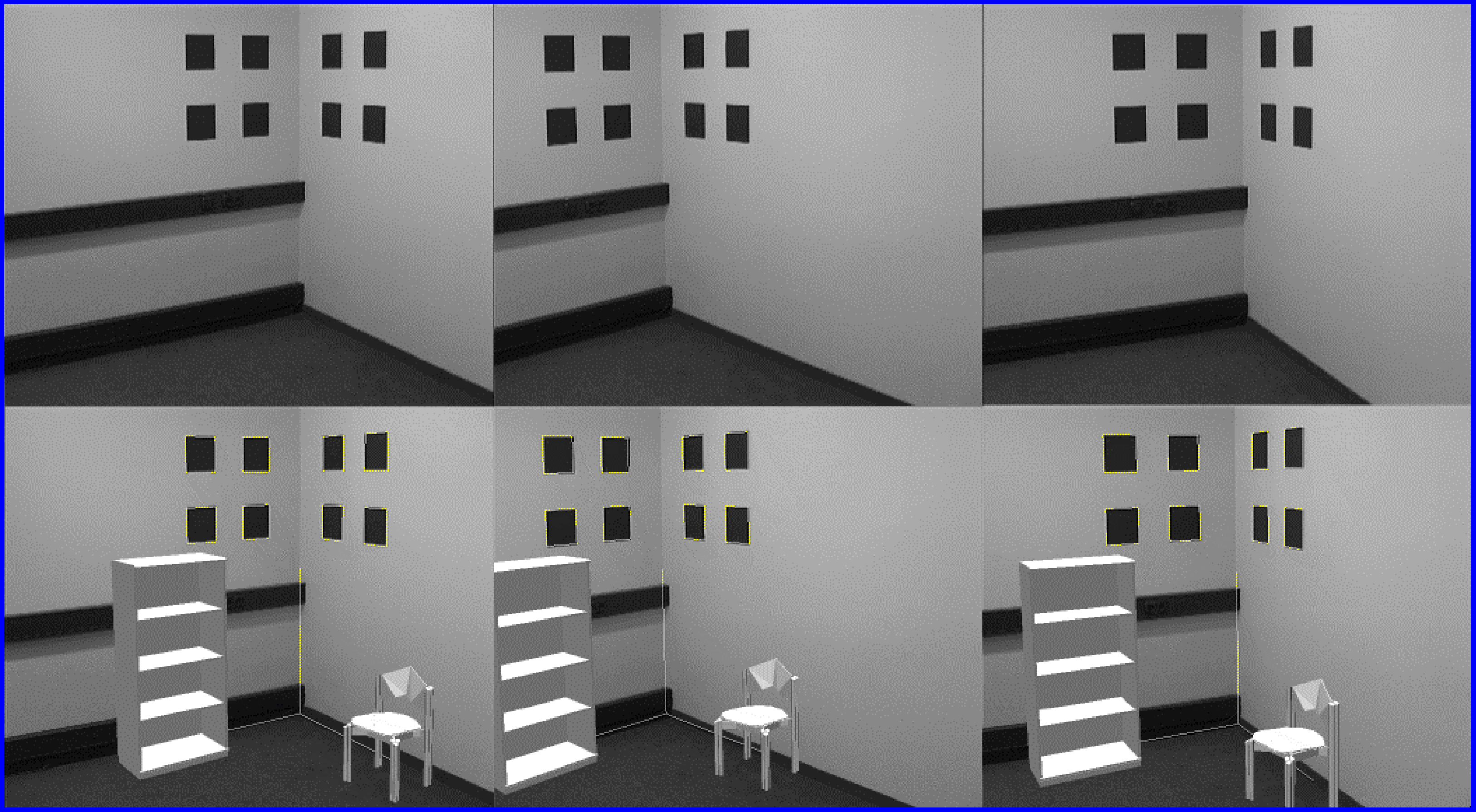
$$\dot{p} = v_c + \omega \times (p - c)$$

where c is the center of mass of the object, p is a point on the object, v_c is the translational velocity at c, and ω is the angular velocity around an axis through c.

Vision Based Camera Tracking

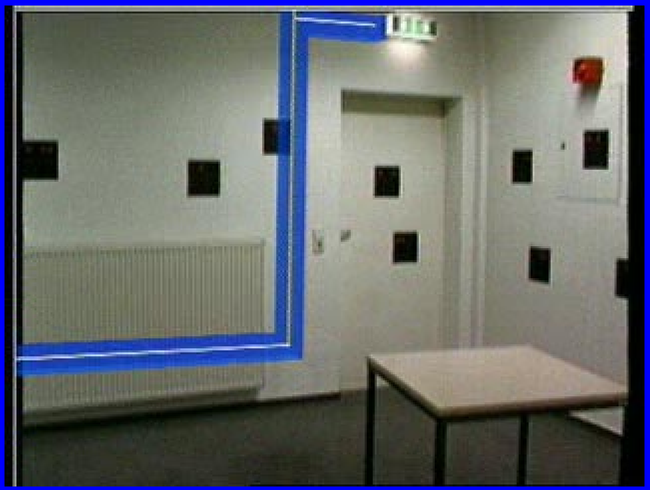
- Extended Kalman filter employed for optimal pose and motion estimation.

Vision Based Camera Tracking Results



Vision Based Camera Tracking

- Example video clip showing camera tracking with augmentation of the scene



Vision Based Camera Tracking

- Occlusions of landmarks tolerated as long as at least two landmarks are completely visible.

Vision Based Tracking

- Objects can also be tracked using landmarks on them that can be detected automatically.
- Other vision-based tracking methods may also be used such as optical flow.

Hybrid Tracking

- Hybrid methods for tracking have been developed for augmented reality
  - UNC research: combination of magnetic trackers and vision based landmark trackers
  - Foxlin: combining inertial trackers with ultrasound.

Interaction of Real and Virtual Objects

- Occlusions
- Interactive queries (e.g., 3D pointer)
- Collisions
- Shadows and lighting effects

Interactions of Real and Virtual Objects: Occlusion

- Method 1: model-based
  - registering a model with the physical object
  - rendering the object in black so graphics can be chroma-keyed
  - use z-buffer to occlude

Interaction of Real and Virtual Objects: Occlusion

- Method 2: geometry extraction
  - compute dense depth map
  - use the computed depth map in z-buffer to occlude virtual objects.

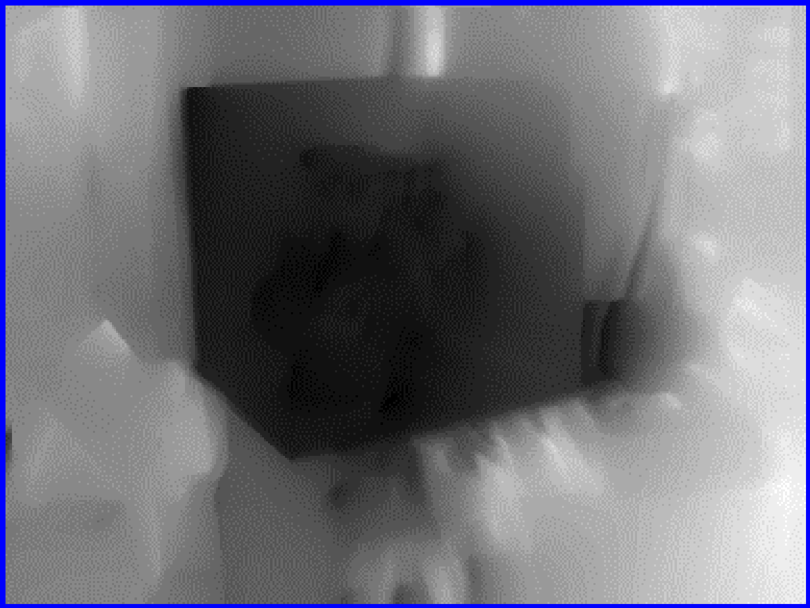
Interaction of Real and Virtual Objects: Occlusion

- Dense depth map can be computed using stereopsis (or other range sensors)

Left Image

Right Image

Interaction of Real and Virtual Objects: Occlusion



Computed Dense Depth Map (darker $\Longrightarrow$ closer)

Depth Map used as z-buffer to obtain occlusion effect

Interaction of Real and Virtual Objects: Collision

- A synthetic object placed in a real scene must give the perception of physical interaction with its environment $\Longrightarrow$ collision detection is an important problem.

Interaction of Real and Virtual Objects: Collision

- Requirements:
  - geometry of the synthetic objects is known
  - geometry of the scene is known
- Classical methods of collision detection can be utilized based on geometry intersection.

  Ref: John Canny

  $\Longrightarrow$ Time consuming

Interaction of Real and Virtual Objects: Collision

- Method 2: use z-buffer techniques
  - Some applications are sufficiently constrained that they can cheat and still have a realistic effect.
  - Scene geometry can be used to build a z-buffer, which then can be used to check for simple collision tests with the synthetic objects.
  - Ref: Breen et al.

Illumination

- Realistic mixing of synthetic objects with images of the scene also involve:
  - Rendering the synthetic objects with the same illumination as the actual scene $\Longrightarrow$ extract illumination sources in the scene
  - Having objects cast shadows on each other in a correct manner $\Longrightarrow$ extract illumination as well as geometry of the scene.

Illumination

- The effect of illumination is summarized by the equation:

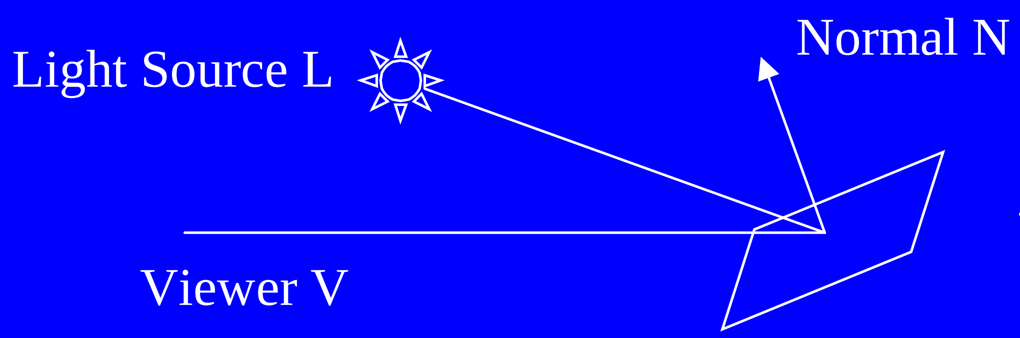
$$L_{pixel}(c) = k_a(c)L_{ia}(c)\pi + k_d(c)\int_{\omega} L(c)(\overrightarrow{N} \cdot \overrightarrow{L})d\omega + k_s(c)\int_{\omega} L(c)(\overrightarrow{N} \cdot \overrightarrow{H})^n d\omega$$

where $c \in \{R, G, B\}$ is the color channel

Illumination

- Most illumination estimation methods utilize some form of shape-from-shading formulation from the diffuse component of the previous equation.

Illumination



$$I = \rho\lambda(\overrightarrow{N} \cdot \overrightarrow{L})$$

- ρ : albedo of the surface
- λ : flux per unit area perpendicular to incident rays
- Ref: Pentland PAMI 1984

Illumination

- In a local homogeneous area of the surface, albedo and illumination change little
- This assumption gives us:

$$
\begin{aligned}
dI &= d(\rho\lambda(\overrightarrow{N} \cdot \vec{L})) \\
&= \rho\lambda(d\overrightarrow{N} \cdot \vec{L}) + \rho\lambda(\overrightarrow{N} \cdot d\vec{L}) \\
&= \rho\lambda(d\overrightarrow{N} \cdot \vec{L})
\end{aligned}
$$

Illumination

- Assumption: changes in surface orientation are isotropically distributed.
- Look at the effect of illuminant direction on $\overline{dI}$, the mean value of *dI*.

$$\overline{dI} = \rho\lambda(\overline{d\vec{N}} \cdot \vec{L})$$
$$= \rho\lambda(x_L d\overline{x}_N + y_L d\overline{y}_N + z_L d\overline{z}_N)$$

$(d\overline{x}_N, d\overline{y}_N, d\overline{z}_N)$: mean change in $d\vec{N}$ measured along direction (dx, dy)

(x_L, y_L, z_L) : light source direction

Illumination

- Assuming change in surface normal is isotropic, then

  $d\bar{x}_N$ is proportional to dx

  $d\bar{y}_N$ is proportional to dy

  $d\bar{z}_N = 0$

  and

$$d\bar{I} = k(x_L dx + y_L dy)$$

Illumination

We can estimate illumination direction from the last equation

- ❶ by measuring the mean of dI along a number of directions
- ❷ and setting up a linear regression formulation to estimate light source direction

Illumination

- Integrated methods for estimating illumination direction with shape-from-shading also exist
  - Ref: Hougen + Ahuja, ICCV 93, Berlin
- More recent works of illumination extraction:
  - Ref: Walter, Alppay, et al, Siggraph '97

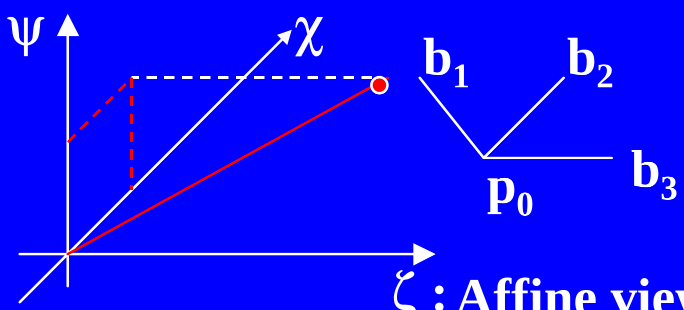
Illumination and Interaction of Real and Virtual Objects: Shadows

- Shadows cast by real objects on virtual objects and vice versa increase the realism in some applications
  - scene and object geometry should be known
  - illumination model should be known
  - then by rerendering the scene one can compute the shadows in the mixed reality scene
  - Ref: Fournier

Calibration-free Augmented Reality

- Display and overlay of graphics is done without extraction of camera parameters.
- Basic operation is reprojection:
  - Given: Four or more 3D points,
    - » the projection of all the points in the set can be computed as a linear combination of the projection of just four of the points.
- Ref: Kutulakos & Vallino in TVCG Jan-March 1998.

Calibration-free Augmented Reality



Affine bases points + p_0 are the fiducial points tracked that define the affine representation

ζ : Affine viewing direction

- Reprojection property

Affine coordinates of p on an object

$$\begin{bmatrix} u \\ v \\ w \\ 1 \end{bmatrix} = \begin{bmatrix} u_{b_1} - u_{p_0} & u_{b_2} - u_{p_0} & u_{b_3} - u_{p_0} & u_{p_0} \\ v_{b_1} - v_{p_0} & v_{b_2} - v_{p_0} & v_{b_3} - v_{p_0} & v_{p_0} \\ & \varsigma^T & & z_{p_0} \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Calibration-free Augmented Reality

- Affine coordinates of a 3D point p can be computed from their projections along two different viewing directions.

$$\begin{bmatrix} u_p^1 \\ v_p^1 \\ u_p^2 \\ v_p^2 \end{bmatrix} = \begin{bmatrix} u_{b_1}^1 - u_{p_0}^1 & u_{b_2}^1 - u_{p_0}^1 & u_{b_3}^1 - u_{p_0}^1 & u_{p_0}^1 \\ v_{b_1}^1 - v_{p_0}^1 & v_{b_2}^1 - v_{p_0}^1 & v_{b_3}^1 - v_{p_0}^1 & v_{p_0}^1 \\ u_{b_1}^2 - u_{p_0}^2 & u_{b_2}^2 - u_{p_0}^2 & u_{b_3}^2 - u_{p_0}^2 & u_{p_0}^2 \\ v_{b_1}^2 - v_{p_0}^2 & v_{b_2}^2 - v_{p_0}^2 & v_{b_3}^2 - v_{p_0}^2 & v_{p_0}^2 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Solve for x, y, and z

Known from coodinates in two images

Affine coordinates

Calibration-free Augmented Reality

- The affine representation of a 3D object can be used by tracking 4 fiducial points across frames and using their image coordinates to compute reprojection of the other points on the objects.

Future Research Topics

- Calibration of see-through HMD’s
  - Optics are different (human eyes are in the loop)
  - There is no explicit image from which points are to be picked directly.
  - Calibration must be done using some other, possibly interactive technique.

Future Research Topics

- Increased accuracy and automation in
  - object registration, and
  - tracking (camera and/or object)
- New display technologies
  - for example, projection of graphics onto real scenes

Resources

- My Web page address:

  http://www.cs.iupui.edu/~tuceryan/AR/AR.html
- Augmented Reality Page by Vallino

  http://www.cs.rochester.edu/u/vallino/research/AR

  Many links to other places from here.
- Fraunhofer AR page:

  http://www.igd.fhg.de/www/igd-a4/ar/
- MIT AI Lab image guided surgery page:

  http://www.ai.mit.edu/projects/vision-surgery/surgery_home_page.html