Module IN 2018

Introduction to Augmented Reality

Prof. Gudrun Klinker
with contributions from M. Huber, D. Pustka

Display Calibration

SS 2018

Literature

- "Single-point active alignment method (SPAAM) for optical see-through HMD calibration for augmented reality", M. Tuceryan, Y. Genc and N. Navab. Presence: Teleoperators and Virtual Environments, Vol. 11, Issue 3, pp. 259-276, June 2002.
- "Display-Relative Calibration for Optical See-Through Head-Mounted Displays", C. Owen, J. Zhou, A. Tang and F. Xiao, in Proceedings of the IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR), 2004
- "Practical Solutions for Calibration of Optical See-Through Devices", Y. Genc, M. Tuceryan and N. Navab, in Proceedings of the IEEE International Symposium on Mixed and Augmented Reality (ISMAR), 2002
- "Real time Versatile Robotics Hand/Eye Calibration using 3D Machine Vision", R. Tsai and R. Lenz, in IEEE International Conference on Robotics and Automation, 1988

Agenda

1. Introduction
2. Video see-through calibration
3. Optical see-through calibration
4. Summary

Reminder

Many kinds of display devices for AR

- Portable displays
  - PDAs, Smartphones
  - Laptops, Tablets
- Projectors
- etc.

Today: only head mounted displays (HMDs)

- Video see-through
- Optical see-through

Motivation

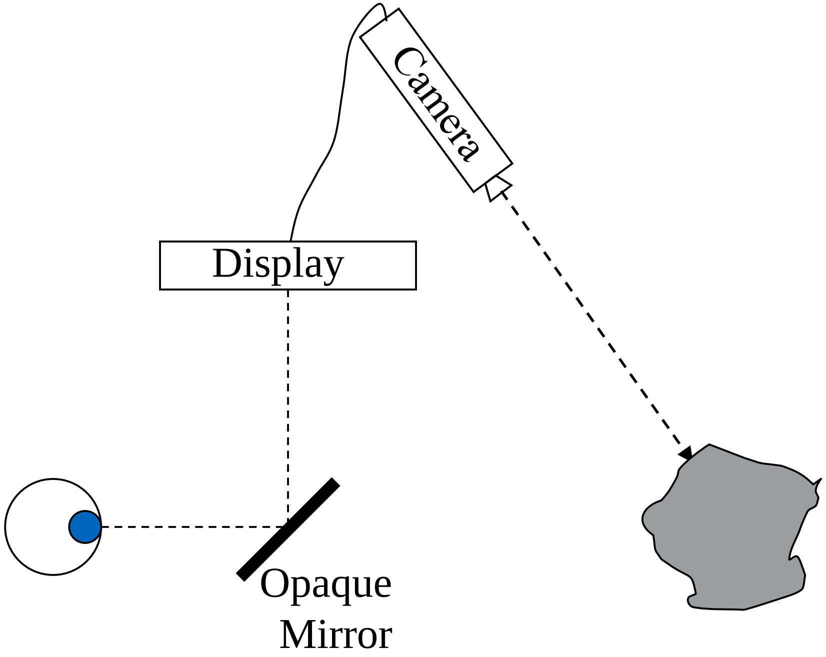
+

=

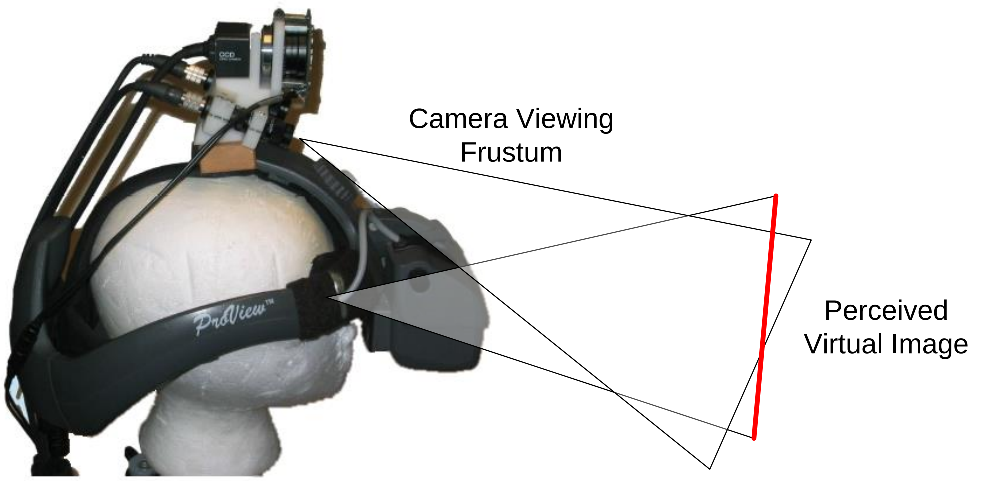
Agenda

1. Introduction
2. Video see-through calibration
3. Optical see-through calibration
4. Summary

Reminder: Video See-Through HMD



Video See-Through: Model



- Problem: Perceived viewpoint is always the same as the camera
  - Try to move camera as closely as possible to eyes
  - Choose camera with approx. same viewing angle as display

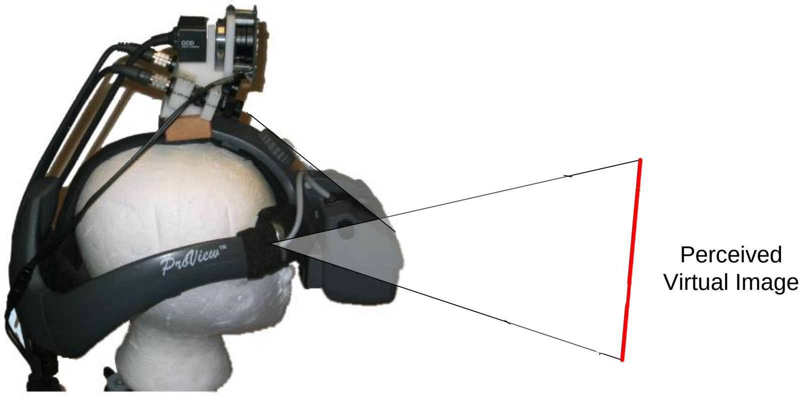
2. Video See-Through Calibration

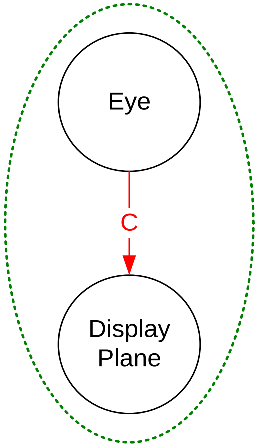
→ 2.1 Using inside-out tracking

2.2 Using outside-in tracking

Based on Video Image

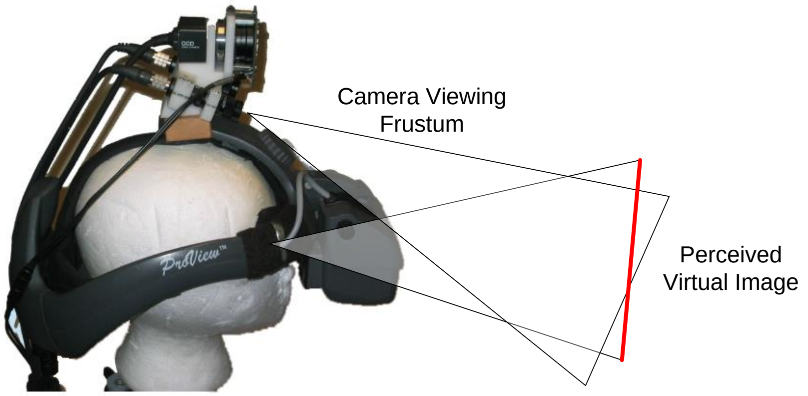
- What projection matrix to load into OpenGL?
- Needed: HMD intrinsics (FOV, etc.) + extrinsics (eye position and orientation)

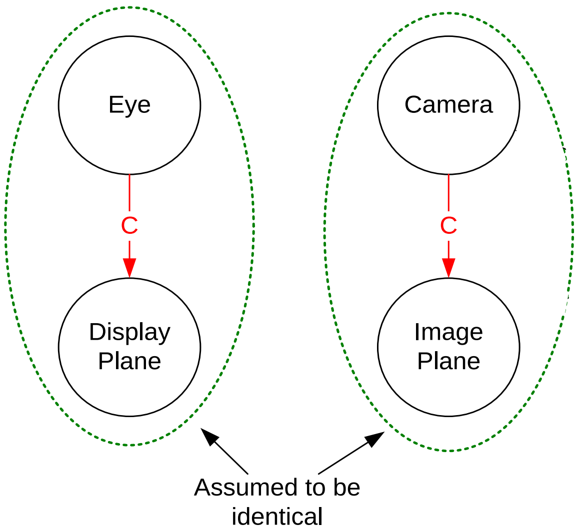




Based on Video Image

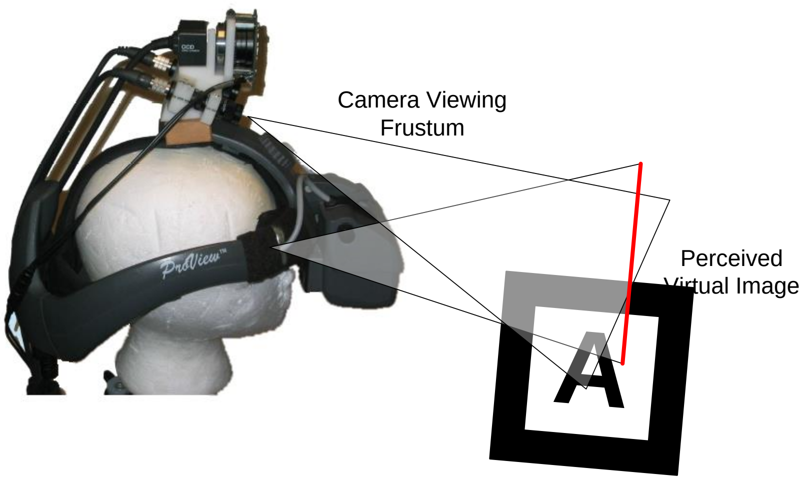
- What projection matrix to load into OpenGL?
- Needed: HMD intrinsics (FOV, etc.) + extrinsics (eye position and orientation)

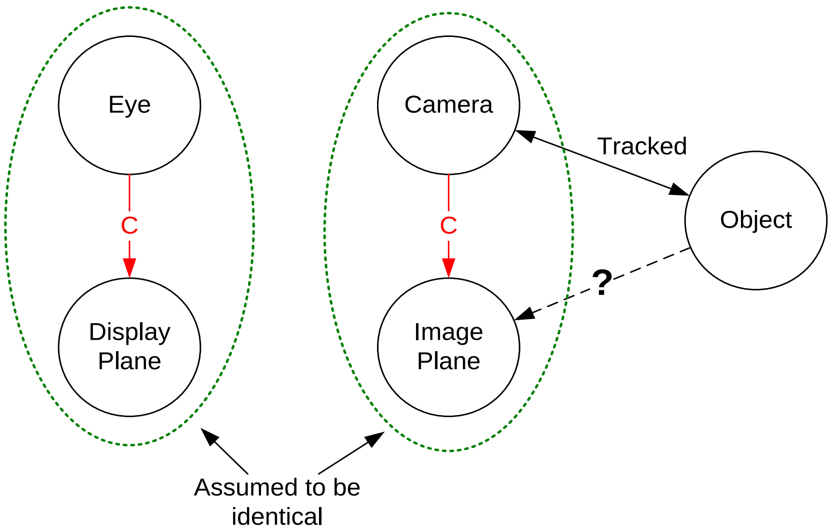




Based on Video Image

- What projection matrix to load into OpenGL?
- Needed: HMD intrinsics (FOV, etc.) + extrinsics (eye position and orientation)





- → Normal camera calibration (e.g. Tsai, Zhang)
- Unwarp radial distortion before displaying!!

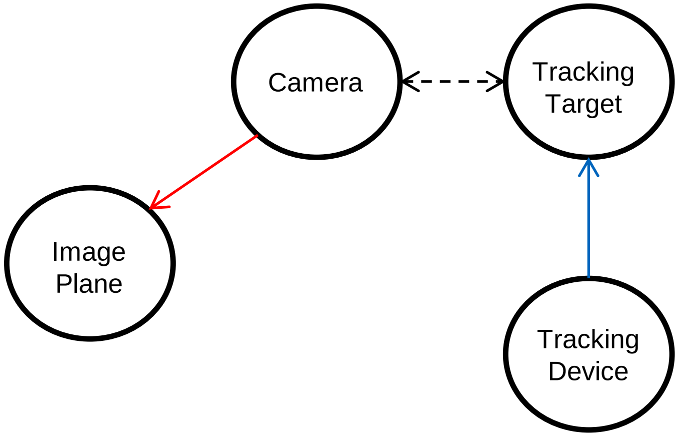
2. Video See-Through Calibration

2.1 Using inside-out tracking

→ 2.2 Using outside-in tracking

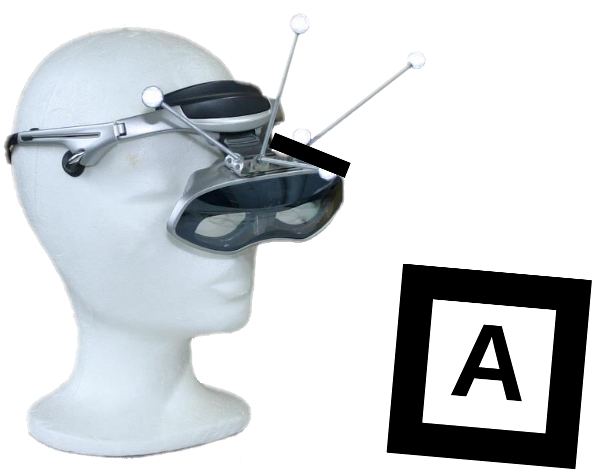
Based on Hand-Eye Calibration

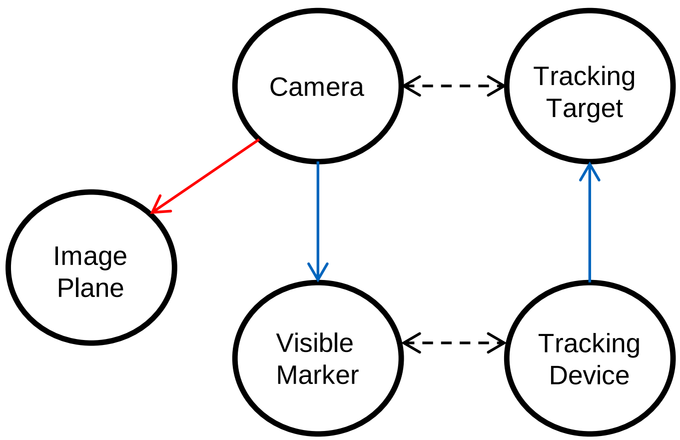
- Problem: no direct link between camera and tracker coordinate systems

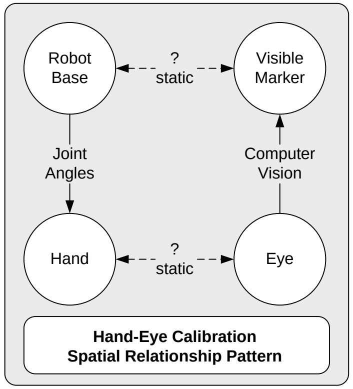


Based on Hand-Eye Calibration

- Idea: use video camera for tracking – but only during calibration
- Using Hand-Eye calibration, the video camera does not need to track the same object as the tracker
  - but transformations must not change!

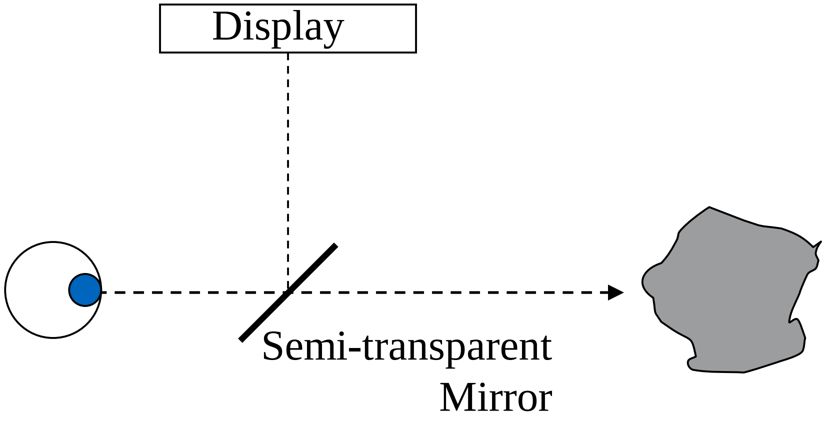






Agenda

Reminder: Optical See-Through HMD

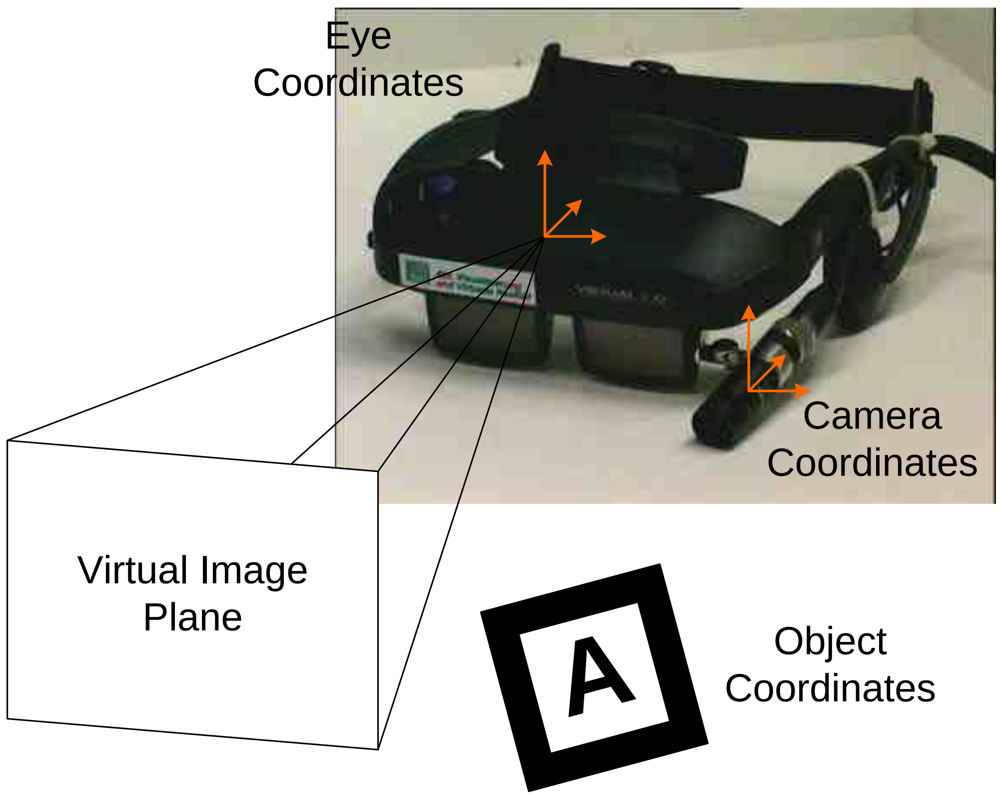


3. Optical See-Through Calibration

→ 3.1 Single Point Active Alignment Method (SPAAM)

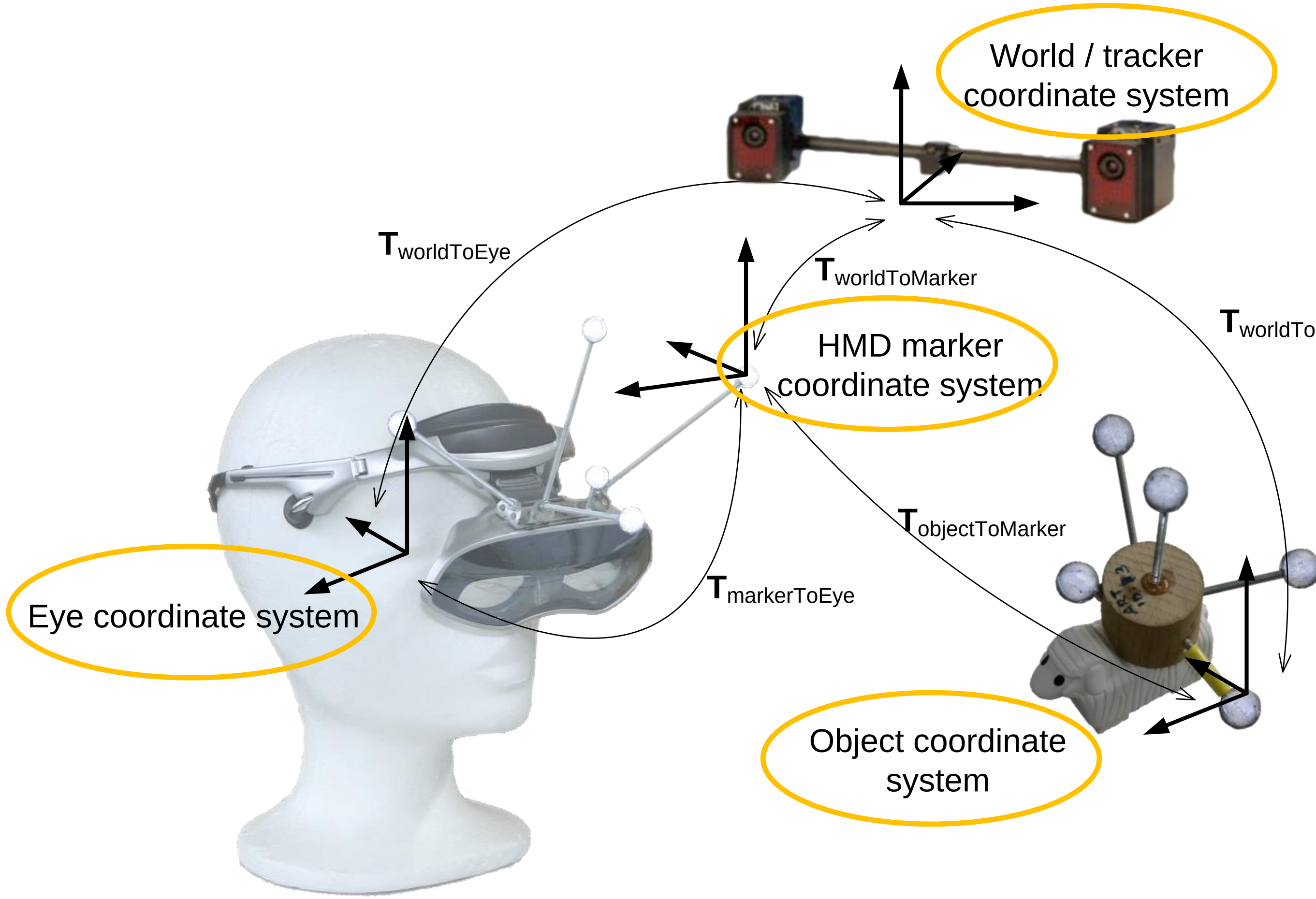
3.2 Display Relative Calibration (DRC)

Using Inside-Out Tracking

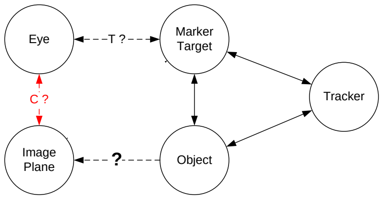


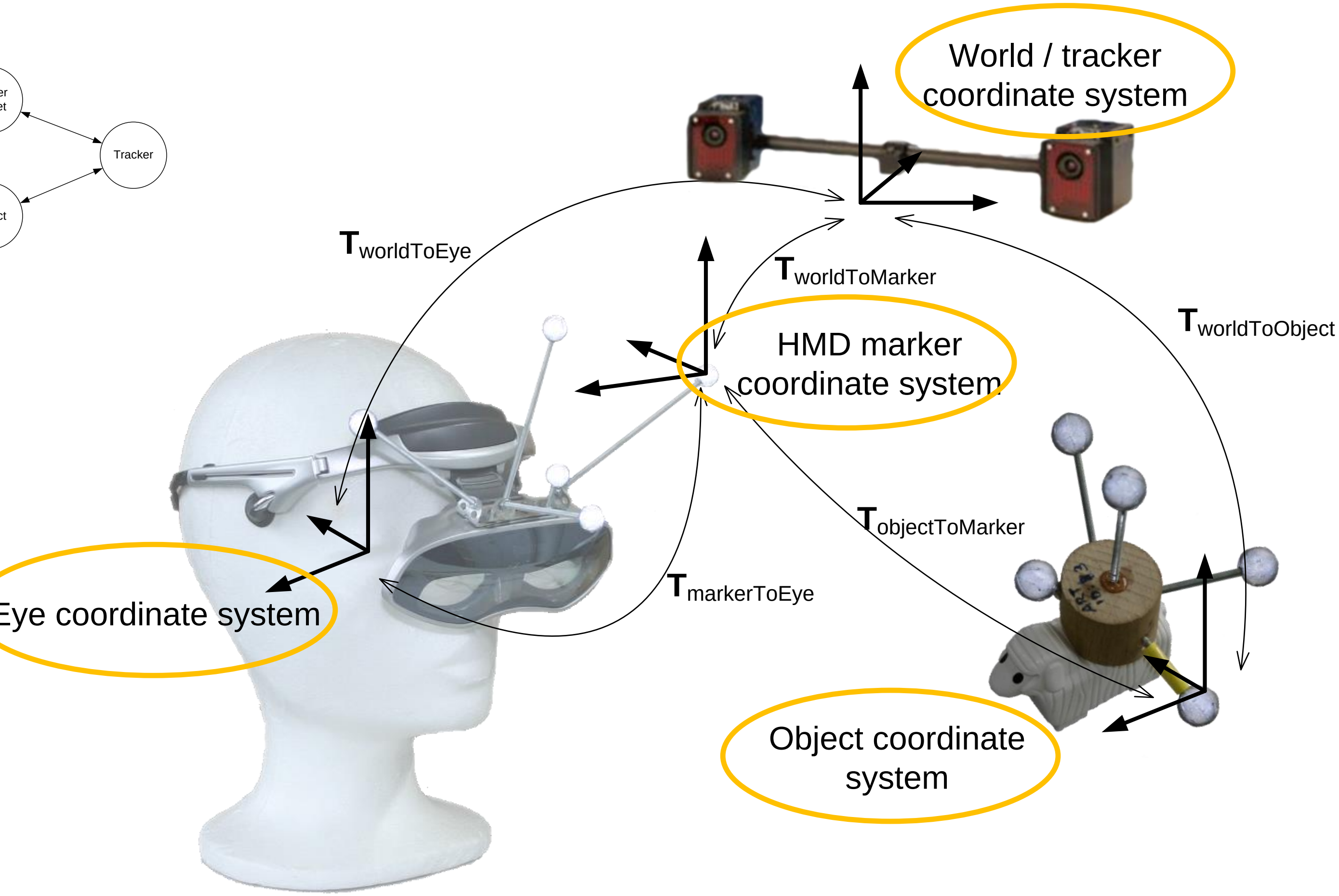
Camera calibration done independently of HMD calibration

Using Outside-In Tracking

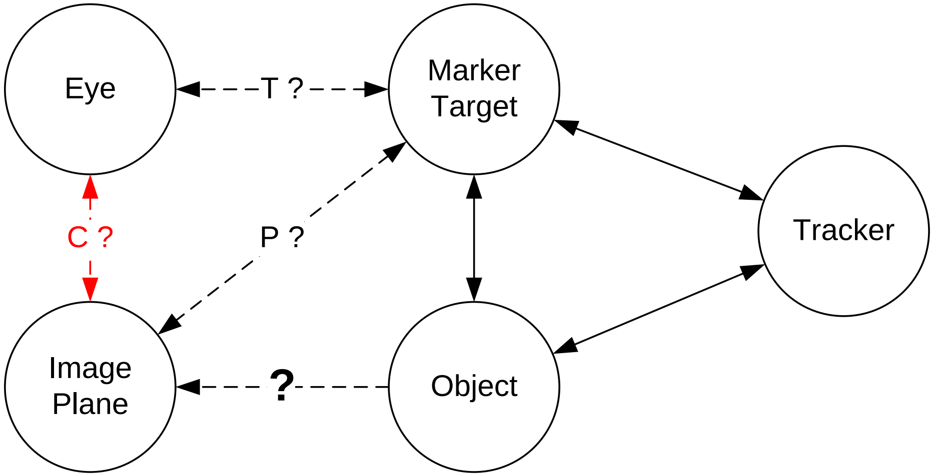


Using Outside-In Tracking



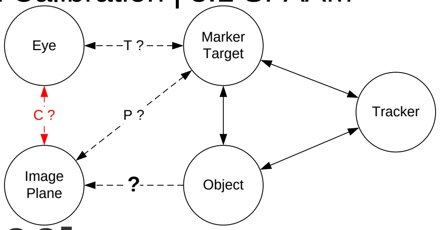


Problem Definition

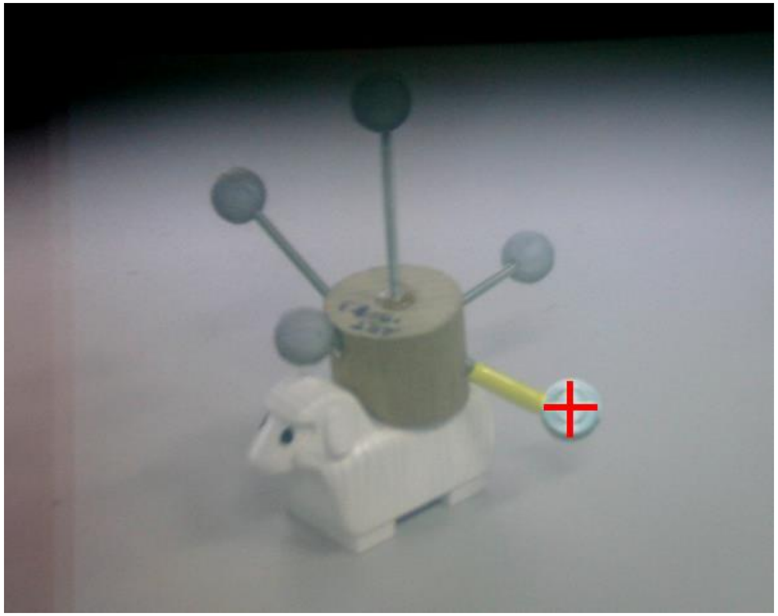


- Unknowns
  - Transformation from marker coordinate frame to viewpoint (translation and orientation): 4x4 matrix T
  - Intrinsic camera parameters: 3x4 matrix C
- If both T and C were known:
  - Projection matrix P = CT

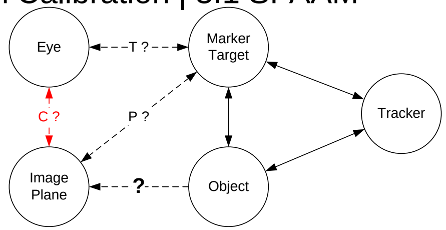
Estimating P Directly



- Single Point Active Alignment Method [Tuceryan 2000]
- Given 3D points in tracker target coordinate frame **and**
  Given **corresponding** 2D points on the screen
  P can be computed directly



Computing P



- Given: Homogeneous 3D points $\mathbf{x}_i = (x_i, y_i, z_i, w_i)^T$ and 2D points $\mathbf{x}'_i = (x'_i, y'_i, w'_i)^T$ s.t.
$$\mathbf{x}'_i = \mathbf{P}\mathbf{x}_i$$
- Where P is a homogeneous 3x4 matrix
- Directly solving above equation not possible, as homogeneous x'_i is defined up to scale
- Idea: Solve $\mathbf{x}'_i \times \mathbf{P}\mathbf{x}_i = \mathbf{0}$ (parallel vectors)
- Equivalent since $\mathbf{a} \times \lambda\mathbf{a} = \mathbf{0}$

Computing P

$$\mathbf{P} = \begin{bmatrix} \mathbf{p}^{1^\top} \\ \mathbf{p}^{2^\top} \\ \mathbf{p}^{3^\top} \end{bmatrix} \qquad \mathbf{x}'_i = (x'_i, y'_i, w'_i)^T \quad \text{with} \quad \mathbf{a} \times \mathbf{b} = \begin{pmatrix} a_2 b_3 - a_3 b_2 \\ a_3 b_1 - a_1 b_3 \\ a_1 b_2 - a_2 b_1 \end{pmatrix}$$

$$\mathbf{x}'_i \times \mathbf{P}\mathbf{x}_i = \begin{pmatrix} y'_i \mathbf{p}^{3^\top} \mathbf{x}_i - w'_i \mathbf{p}^{2^\top} \mathbf{x}_i \\ w'_i \mathbf{p}^{1^\top} \mathbf{x}_i - x'_i \mathbf{p}^{3^\top} \mathbf{x}_i \\ x'_i \mathbf{p}^{2^\top} \mathbf{x}_i - y'_i \mathbf{p}^{1^\top} \mathbf{x}_i \end{pmatrix} = 0$$

$$\left(\mathbf{p}^{1^\top}, \mathbf{p}^{2^\top}, \mathbf{p}^{3^\top}\right)^\top \bullet \quad \text{Factoring for}$$

$$\begin{bmatrix} 0^\top & -w'_i \mathbf{x}_i^\top & y'_i \mathbf{x}_i^\top \\ w'_i \mathbf{x}_i^\top & 0^\top & -x'_i \mathbf{x}_i^\top \\ -y'_i \mathbf{x}_i^\top & x'_i \mathbf{x}_i^\top & 0^\top \end{bmatrix} \begin{pmatrix} \mathbf{p}^1 \\ \mathbf{p}^2 \\ \mathbf{p}^3 \end{pmatrix} = 0$$

SPAAM Algorithm: Computing P

$$\begin{bmatrix} 0^\top & -w_i'\mathbf{x}_i^\top & y_i'\mathbf{x}_i^\top \\ w_i'\mathbf{x}_i^\top & 0^\top & -x_i'\mathbf{x}_i^\top \\ -y_i'\mathbf{x}_i^\top & x_i'\mathbf{x}_i^\top & 0^\top \end{bmatrix} \begin{pmatrix} \mathbf{p}^1 \\ \mathbf{p}^2 \\ \mathbf{p}^3 \end{pmatrix} = 0$$

- the three rows are not linearly independent → last row is usually dropped (provided $w'_i \neq 0$)

$$\begin{bmatrix} 0^\top & -w_i'\mathbf{x}_i^\top & y_i'\mathbf{x}_i^\top \\ w_i'\mathbf{x}_i^\top & 0^\top & -x_i'\mathbf{x}_i^\top \end{bmatrix} \begin{pmatrix} \mathbf{p}^1 \\ \mathbf{p}^2 \\ \mathbf{p}^3 \end{pmatrix} = \mathbf{A}_i\mathbf{p} = 0$$

- Stacking matrices A_i for all correspondences i gives matrix A
- Homogeneous Linear Equation System

SPAAM Algorithm: Solving for P

- We now have $\mathbf{Ap} = \mathbf{0}$, where A is a (n*2)x12 matrix and p is a homogeneous 12-vector (→ defined up to scale)
- Straightforward solving would result in trivial solution $\mathbf{p} = \mathbf{0}$
- Solution is the null-space (kernel) of A
- In this case kernel is 1D subspace
- Null-space: (Right) singular vectors $\mathbf{x}$ corresponding to singular value 0

$$\mathbf{Ax} = 0 \cdot \mathbf{x}$$

- Compute p using Singular Value Decomposition

$$\mathbf{A}_{m\times n} = \mathbf{U}_{m\times m}\mathbf{\Sigma}_{m\times n}\mathbf{V}^{\mathsf{T}}_{n\times n}$$

  where p is the last column of V
- SVD also enforces constraint $\|\mathbf{p}\| = 1$

Creating a Projection Matrix for OpenGL

- P is a 3x4 matrix, OpenGL requires 4x4 projection matrices
- Copy third row and modify to account for depth clipping

$$\mathbf{P}_{\text{OpenGL}} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -z_{\text{near}} - z_{\text{far}} \\ 0 & 0 & 1 \end{bmatrix} \mathbf{P} + \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & z_{\text{near}} \cdot z_{\text{far}} \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

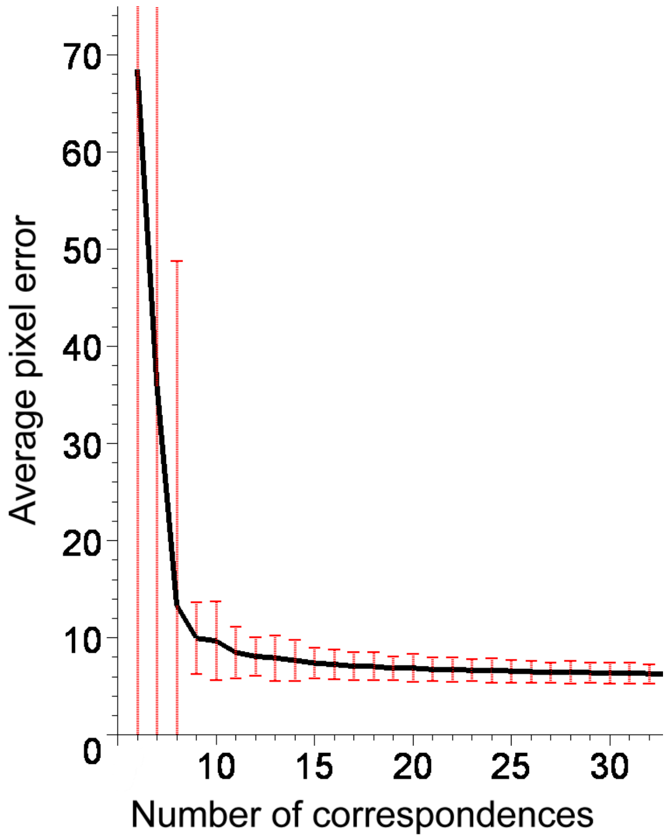
- Requires normalization s.t. $\left\| p_{3,1} \quad p_{3,2} \quad p_{3,3} \right\| = 1$
- Load into OpenGL:

```
glMatrixMode( GL_PROJECTION );
glLoadIdentity();
glOrtho( 0, w, 0, h, z_near, z_far );
glMultMatrixd( P_OpenGL );
```

Considerations for Choosing Points (I)

Minimum number of required point correspondences:

- P has 12 entries, but only 11 degrees of freedom
- Each correspondence gives 2 constraints
  → at least 6 (>5.5) corresponding points required
- User input is noisy
  → good results achieved with
  n >= 15



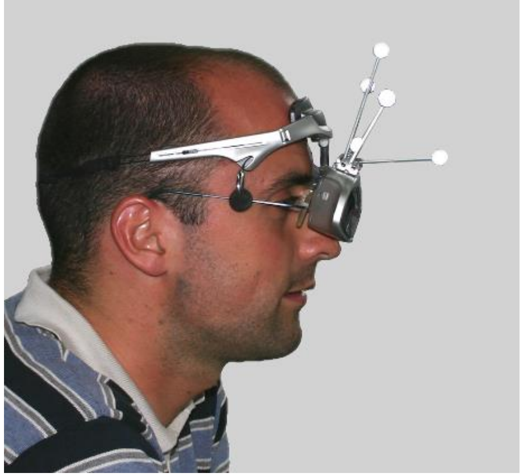
Considerations for Choosing Points (II)

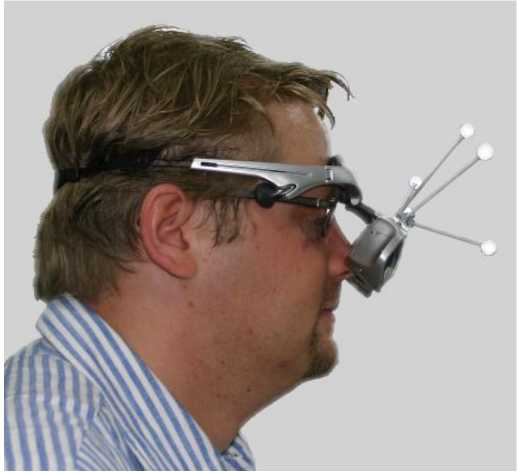
Perspective projection depends on depth

- 3D points must be at different distances from the user
  - Software should instruct user to change distance
  - Software should reject point sets with insufficient depth
- 3D points must not be co-planar
- Ideal: sampling of the whole working volume of the application

How Often is Calibration Necessary?

- Arrangement of tracker and HMD is fixed
- Users have different physiology
  - Size of head
  - Position of eye
  - Interpupillary distance
- Different ways of putting on the HMD
- Calibration is necessary
  - for every user
  - when HMD is moved





Calibrating a Stereo HMD

- Either calibrate the two displays individually
  - Drawback: requires >2*15 interactions
- Or use Stereo-SPAAM
  - Show crosshair in both displays simultaneously
    → crosshair also has a 3D position
  - User has to align crosshair with reference point in 3D
    → creates valid correspondences for both displays
  - Then use normal matrix estimation to estimate P_L and P_R separately
  - Drawback: correct alignment in 3D is difficult

Refining Existing Calibrations: EasySPAAM

- Having to provide >15 points every time the user puts on the HMD is tedious
- Idea: Add correction to existing SPAAM calibration
- Correction modeled by warping in 2D on the image plane

$$\mathbf{P}' = \mathbf{S}\mathbf{P}$$

$$\mathbf{S} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \quad \text{or} \quad \mathbf{S} = \begin{bmatrix} s_x & 0 & t_x \\ 0 & s_y & t_y \\ 0 & 0 & 1 \end{bmatrix}$$

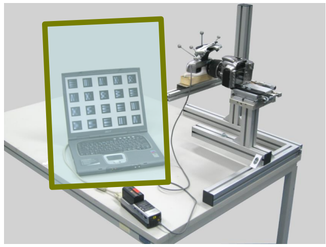
- Simpler model with less parameters → less points required
- [Tuceryan 2002b]

3. Optical See-Through Calibration

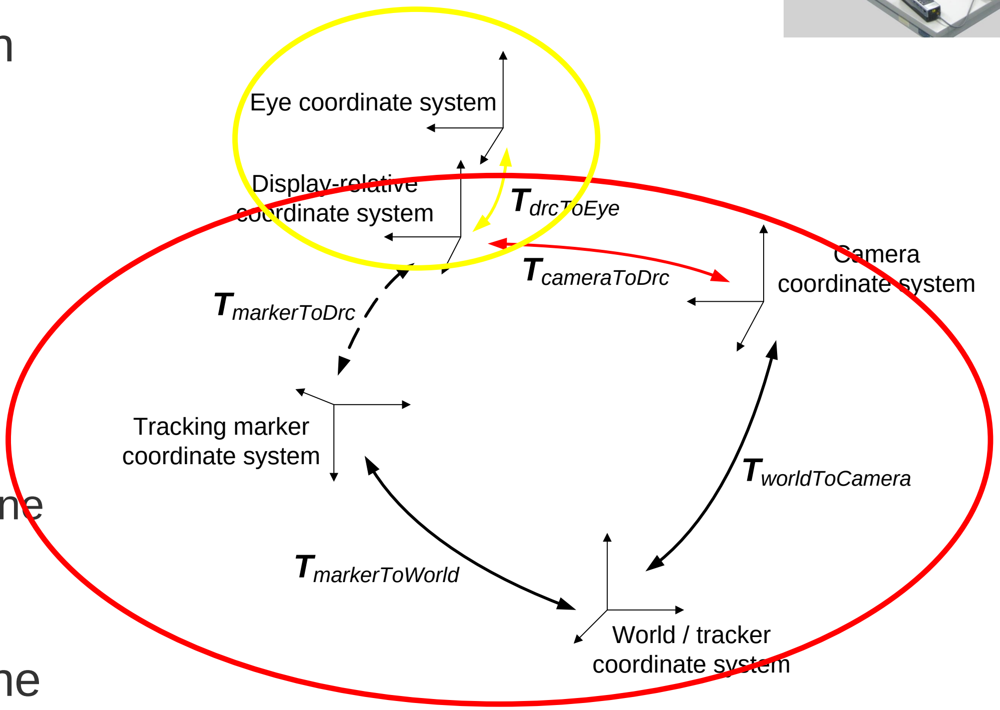
3.1 Single Point Active Alignment Method (SPAAM)

→ 3.2 Display Relative Calibration (DRC)

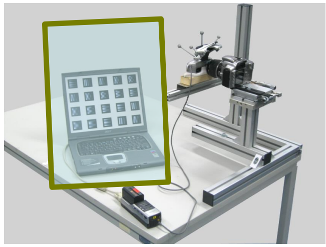
Advanced Image Formation Model



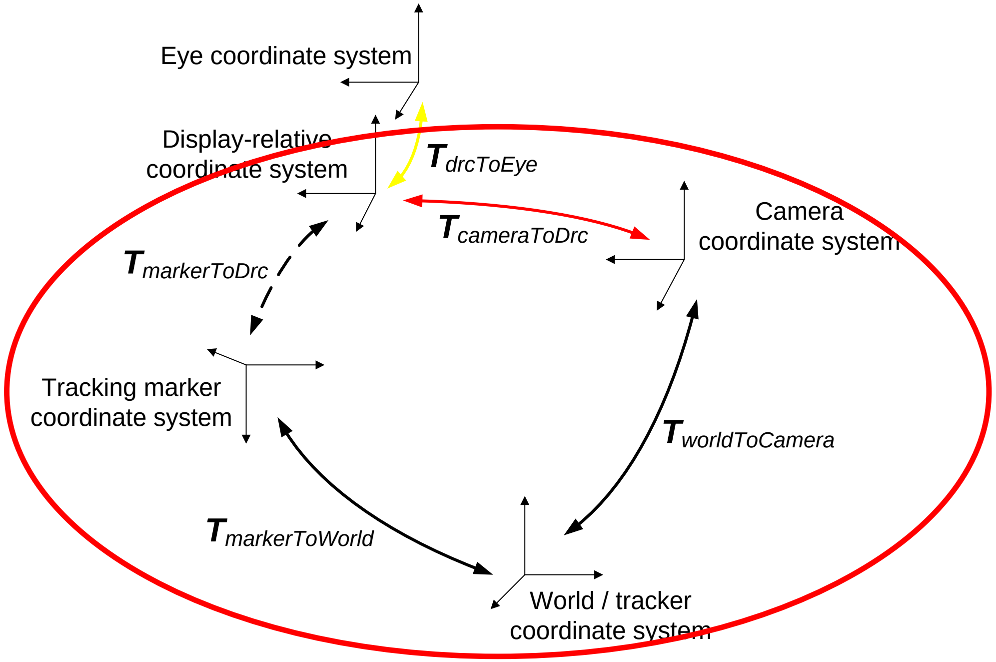
- If HMD and marker are fixed, only the eye position changes
  - less parameters to estimate (3 vs. 11) → less points
  - Two-phase approach
    - Offline: replace eye with camera
    - Online: only determine eye position
- SPAAM does not take the position of the display plane into account
- [Owen 2004]



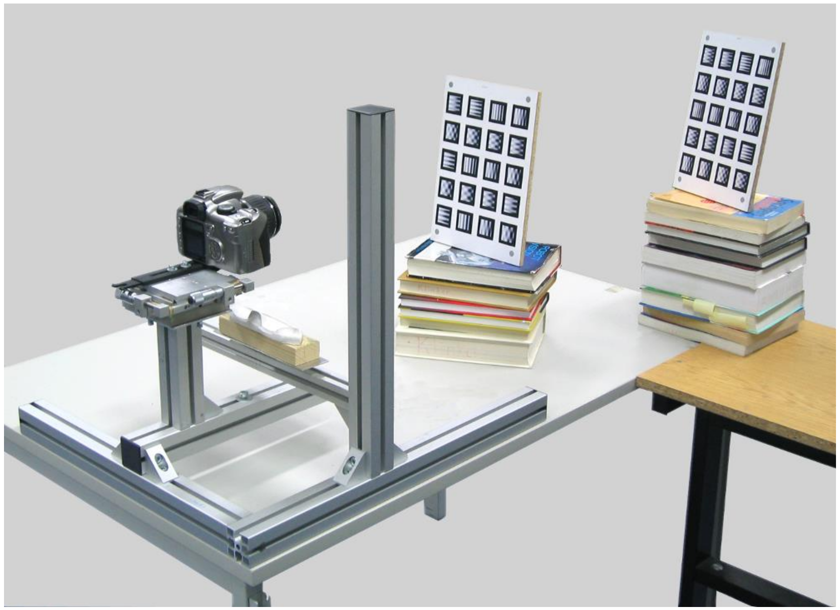
DRC: Offline-Phase



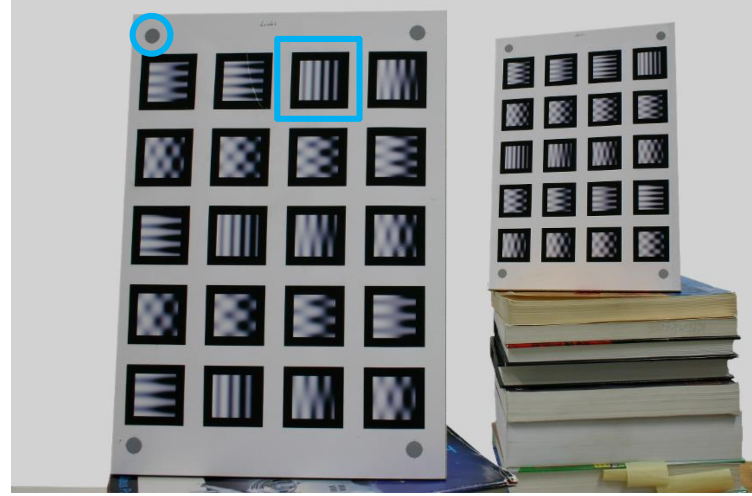
- Determine HMD extrinsic ($T_{markerToDrc}$) and intrinsic parameters
- $T_{markerToDrc} = T_{cameraToDrc} T_{worldToCamera} T_{markerToWorld}$
- Determine position of the display plane



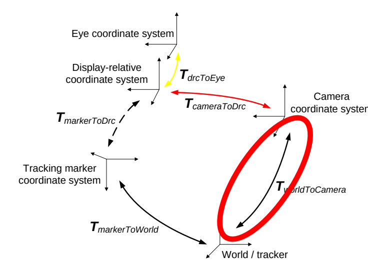
Offline Step 1: Calibrating the Camera



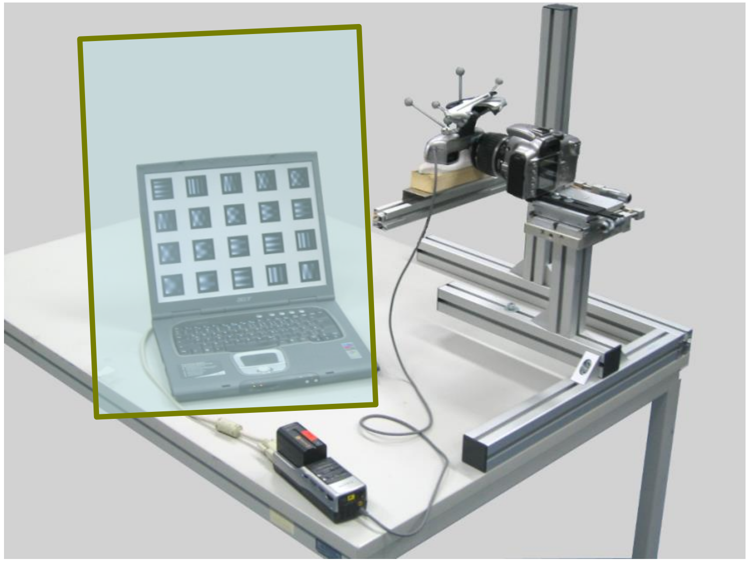
Two sets of optical markers on the same board:
- Squares for Tsai camera calibration
- Circles for ART tracking

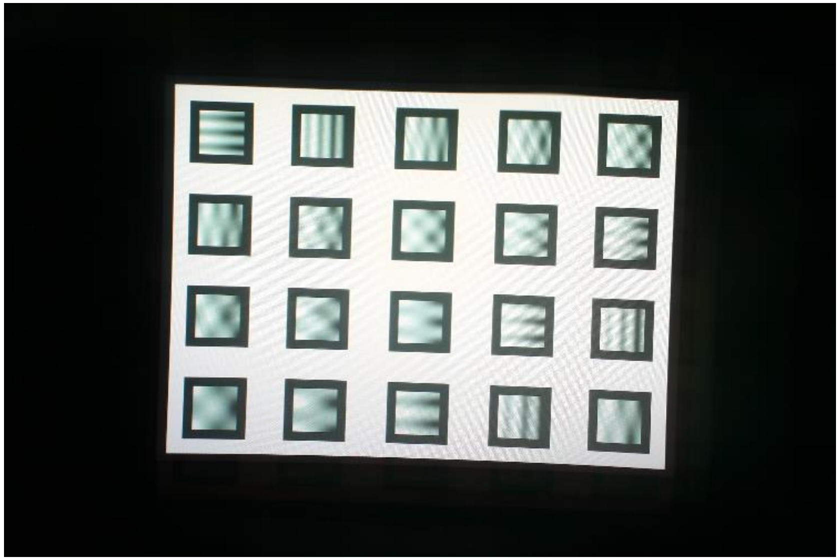


- Calibration patterns are tracked
- Camera calibration (Tsai) computes both $T_{worldToCamera}$ and intrinsic parameters from 2D-3D point sets

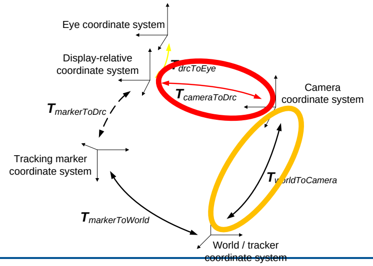


Offline Step 2: Calibrating the HMD

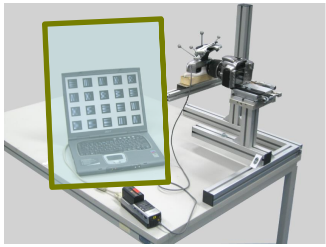




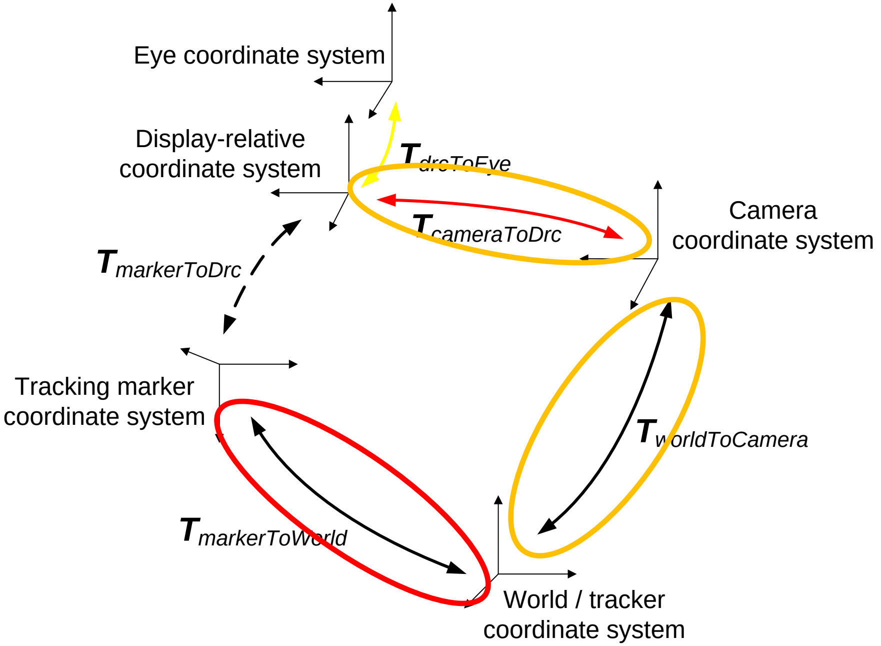
- Find markers projected into the HMD (turned into “opaque” mode)
- Project 2D points into 3D using known camera calibration and arbitrary (but different) depths
- Apply Tsai calibration to compute $T_{cameraToDrc}$ and HMD intrinsics



Offline Step 3: HMD Tracker Pose

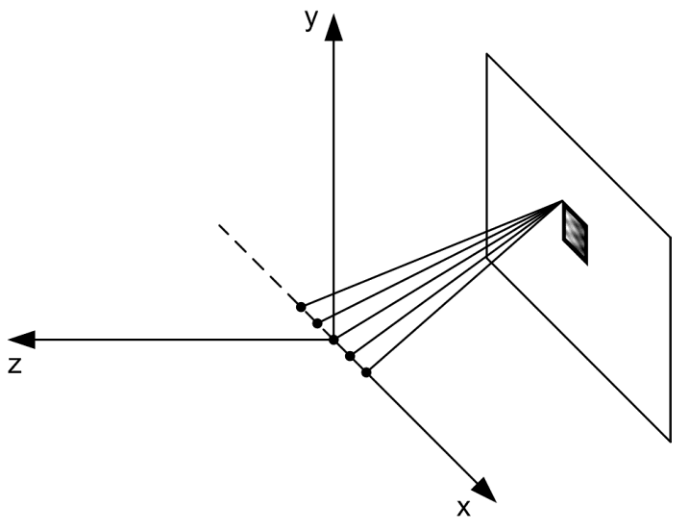


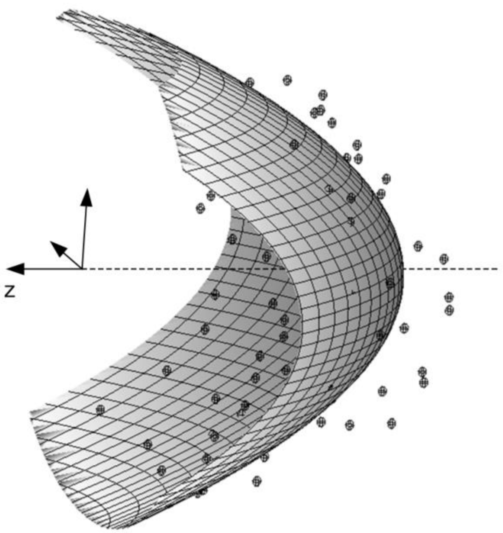
- Trivially done by reading tracker data $T_{markerToWorld}$
- $T_{markerToDrc}$ can now be computed as $T_{cameraToDrc} T_{worldToCamera} T_{markerToWorld}$



Offline Step 4: Compute HMD Image Plane

- Move camera using precision translation stage (several mm)
- Stereo vision to find actual 3D position of points on image plane





Result: Image plane actually is parabolic

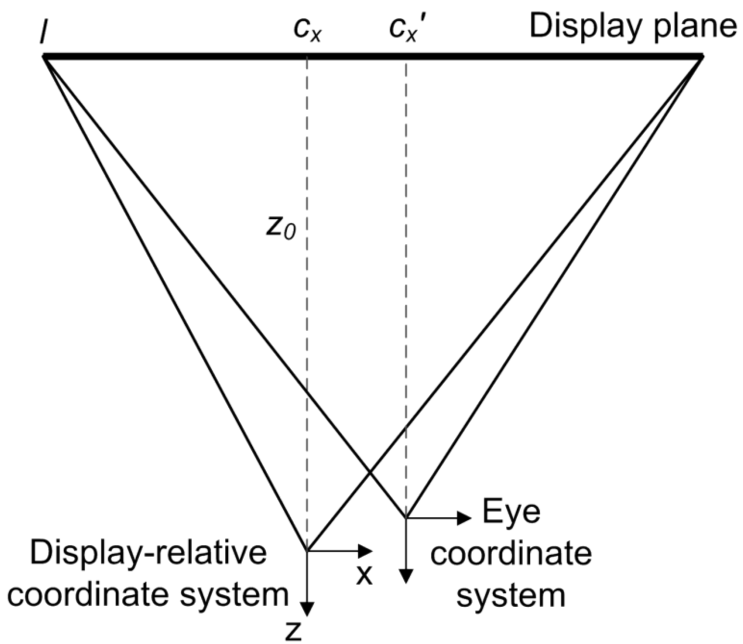
Online Phase

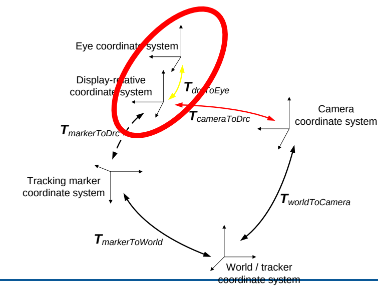
- Determine user-dependent parameter: $T_{drcToEye}$
- Used to compute $T_{markerToEye} = T_{drcToEye}\ T_{markerToDrc}$
- Modification of intrinsic parameters necessary

$$c'_x = c_x - \frac{x_{\mathrm{eye}} f s_x}{z_0}$$

$$c'_y = c_y - \frac{y_{\mathrm{eye}} f}{z_0}$$

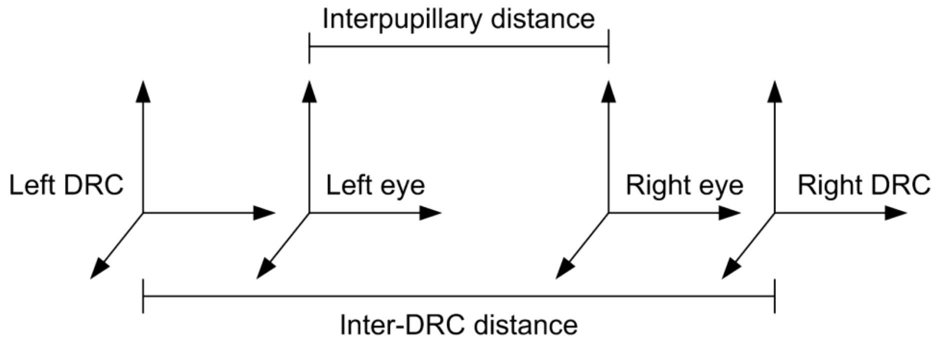
$$f' = f \cdot \frac{z_0 - z_{\mathrm{eye}}}{z_0}$$



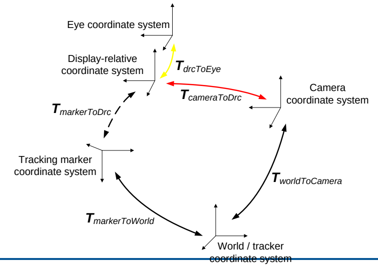


Getting the Eye Position (I)

- Interpupillary distance (for stereo HMDs)
  - Measured using pupilometer
  - Shift eye position in X

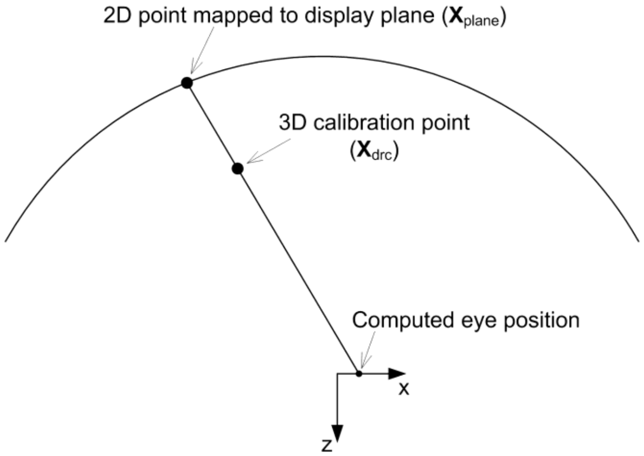


- Manual adjustment
  - using keyboard, joystick, mouse…
  - possible as only three degrees of freedom

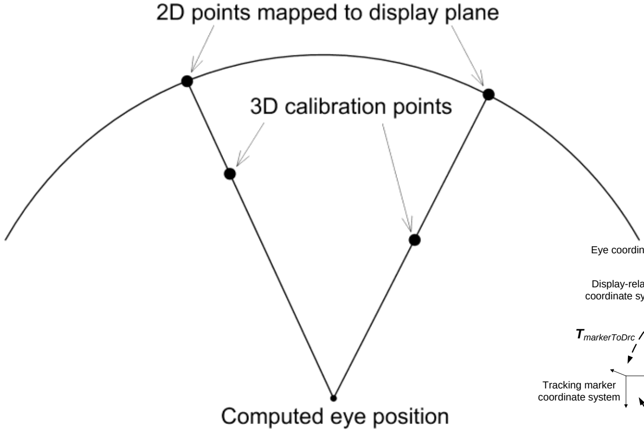


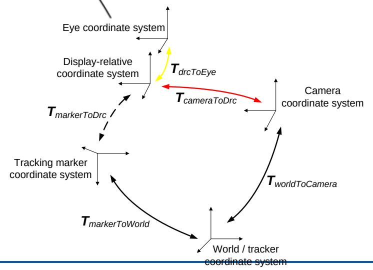
Getting the Eye Position (II)

- Single 2D-3D point correspondence
  - yields eye position in x and y only



- Multiple 2D-3D point correspondences
  - full eye position
  - intersecting rays





Agenda

1. Introduction
2. Video see-through calibration
3. Optical see-through calibration
4. Summary

Summary

- Video see-through displays
  - Inside-Out tracking using see-through camera: Camera calibration (e.g. Tsai or Zhang)
  - Outside-In tracking: Hand-eye calibration
- Optical see-through displays
  - Single Point Active Alignment Method (SPAAM)
    - Collect >15 points by simple user interaction
    - Easy to implement
  - Display Relative Calibration (DRC)
    - Offline-phase requires complicated laboratory setup
    - More advanced model
    - In reality, does not produce better results

Thank you!