Tracking for VR and AR

Hakan Bilen

November 17, 2017

Computer Graphics
University of Edinburgh

Slide credits: Gordon Wetzstein and Steven M. La Valle

Overview

- VR and AR
- Inertial Sensors
  - Gyroscopes
  - Accelerometers
  - Magnetometers
- Positional Tracking
  - Inside-out tracking
  - Outside-in tracking

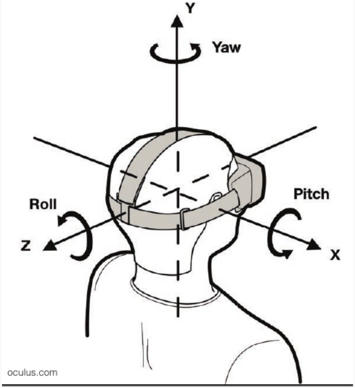
VR and AR

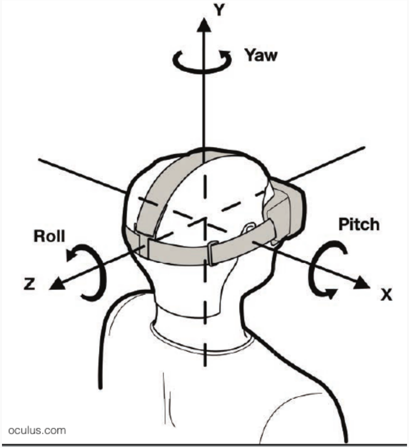
VR and AR

- VR and AR systems should
    - (**VR,AR**) be interactive in real time through multiple sensorial channels (vision, sound, touch, smell, taste)
    - (**AR**) combine real and virtual objects
    - (**AR**) register real and virtual objects
- Registration is one of the most basic problems currently limiting augmented reality

What do we track?

- Goal: Track pose of headset, controller, etc.
- What is a pose?
  - 3D position of the tracked object
  - 3D orientation of the tracked object
- Why? So we can map the movement of our head to the motion of camera in a virtual environment - eg motion parallax.





- Goal: Track pose of headset
- Orientation is the rotation of device w.r.t. world or inertial frame
- $p' = M_{vp} M_{proj} M_{cam} M_m p$
- $M_{cam} = RT$

Inertial Sensors

Inertial Tracking

Modern VR devices

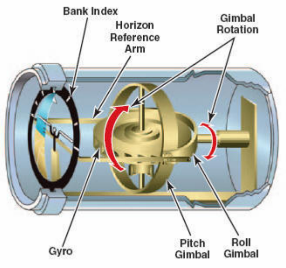
- Oculus Rift has an accelerometer, gyroscope, magnetometer, camera.
- HTC Vive has an accelerometer, gyroscope, Lighthouse laser tracking system, front-facing camera.

What do Inertial Sensors measure?

- gyroscope measures angular velocity $\tilde{\omega}$ in degrees/sec.
- accelerometer measures linear acceleration $\tilde{a}$ in m/sec^2.
- magnetometer measures magnetic field strength $\tilde{m}$ in Gauss (or Tesla).

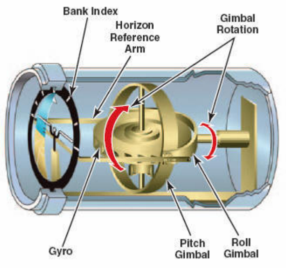
Gyroscopes

- critical for inertial measurements in ballistic missiles, aircrafts, drones, the mars rover, pretty much anything that moves!!
- measures and helps maintaining orientation and angular velocity



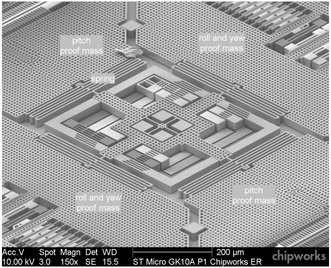
Gyroscopes

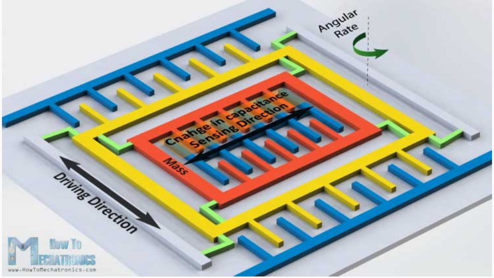
- critical for inertial measurements in ballistic missiles, aircrafts, drones, the mars rover, pretty much anything that moves!!
- measures and helps maintaining orientation and angular velocity
- How? conservation of angular momentum



MEMS Gyroscopes

- today we use microelectromechanical systems (MEMS)
- measures angular rate using the Coriolis Effect





https://www.youtube.com/watch?v=eqZgxR6eRjo

Gyroscopes

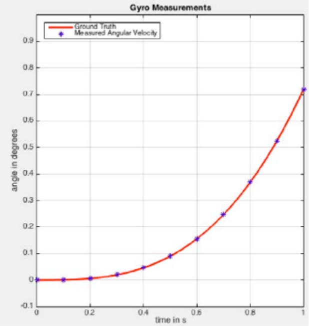
- gyro model: $\tilde{\omega} = a\omega + b + \eta$
  - ω is true angular velocity
  - a is scalar
  - b is bias, temperature dependent but approximated as a const
  - $\eta \approx N(0, \sigma_{gyro}^2)$ is zero-mean Gaussian noise
- 3 DOF
- calibrate:
  - assume that you have a pre-calibrated one $\tilde{\omega}'$
  - minimise $\sum_i^n (\tilde{\omega}_i - \tilde{\omega}'_i)^2$ and find optimal $a^*, b^*, \sigma_{gyro}^*$
  - $\omega_{cal} = a^*\omega + b^* + \eta^*$

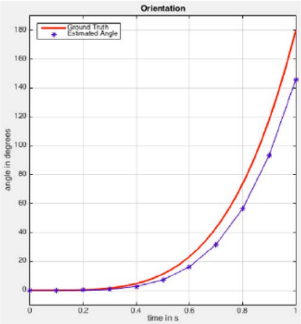
Gyroscopes

- integrate: $\tilde{\theta}[k] = \theta(0) + \sum_i^k \omega_{cal}[i]\Delta t$

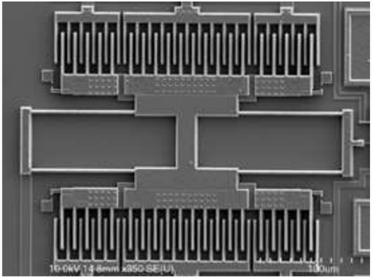
Gyroscopes

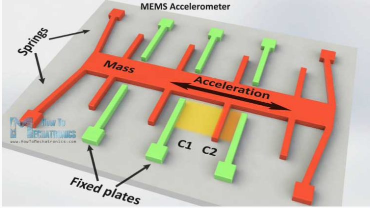
- integrate: $\tilde{\theta}[k] = \theta(0) + \sum_i^k \omega_{cal}[i]\Delta t$
- works well for linear motion
- drift in nonlinear motion
- accurate in short term





- MEMS
- a mass attached to a spring
- acceleration by measuring change in capacitance
- measure linear acceleration: $\tilde{a} = a^{(g)} + a^{(l)} + \eta$
  - $a^{(g)}$ is gravity vector with magnitude 9.81 m/s^2
  - $\eta \approx N(0, \sigma^2_{acc})$ is zero-mean Gaussian noise





Accelerometers

Pros

- points up on average with magnitude of 1g
- accurate in long term, no drift, center of gravity doesn't move

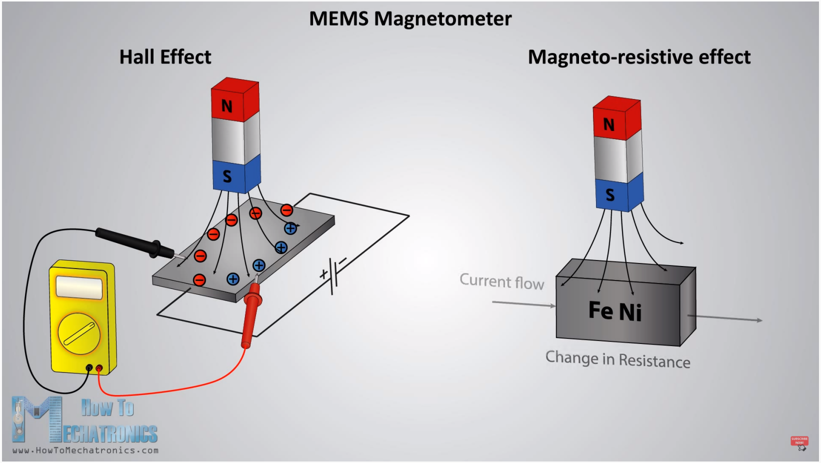
Cons

- noisy measurements
- unreliable in short run due to motion and noise

Complimentary to gyroscope

- fusing gyro and accelerometer gives 6 DOF
- tilt correction (pitch and roll)

Magnetometers



Magnetometers

- measure magnetic field in Gauss or micro Tesla units
- 3 DOF
- Pros
  - together with gyro, accelerometer 9 DOF
- Cons
  - actual direction depends on latitude and longitude
  - distortions due to metal objects

Positional Tracking

Positional Tracking

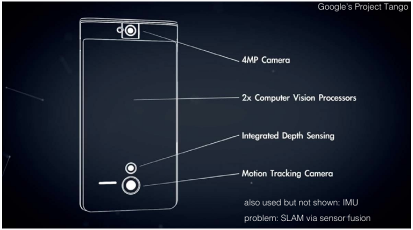
inside-out tracking: camera or sensor is located on HMD, no need for other external devices to do tracking

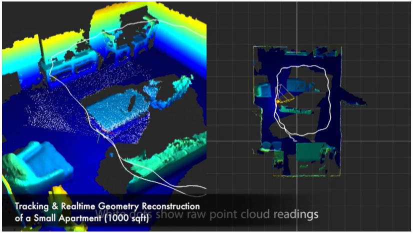
- simultaneous localization and mapping (SLAM) – classic computer & robotic vision problem (beyond this class)

outside-in tracking: external sensors, cameras, or markers are required (i.e. tracking constrained to specific area)

- used by most VR headsets right now, but ultimate goal is insight-out tracking

Inside-out tracking

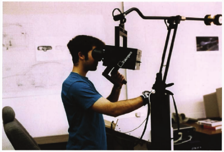


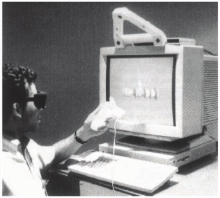


https://www.youtube.com/watch?v=Qe10ExwzCqk

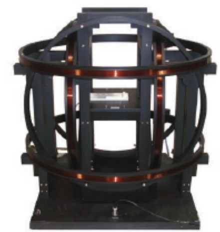
Outside-in tracking

- mechanical tracking
- ultra-sonic tracking
- magnetic tracking
- GPS
- WIFI
- optical





Logitech 6DOF

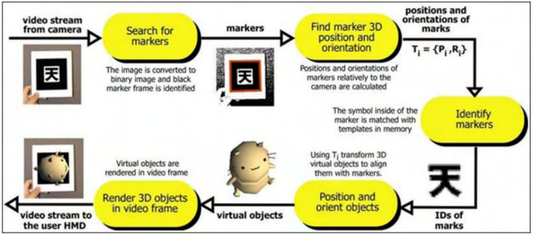


3 axis Helmholtz coil
www.directvacuum.com

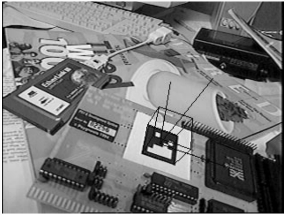
Oculus Rift
https://www.ifixit.com/Teardown/Oculus+Rift+CV1+Teardown/60612

Marker Based Tracking

- Seminal papers by Rekimoto 1998 and Kato & Billinghurst 1999
- ARToolkit and OpenCV+OpenGL



ARToolKit Pipeline



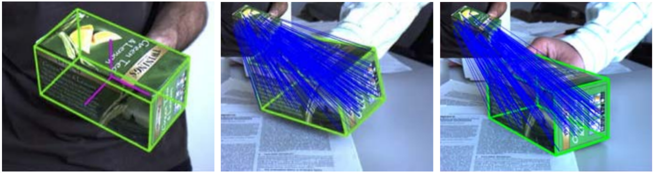
Rekimoto Matrix

Markerless Tracking

- Markers
  - are cheap and robust against lighting changes
  - but do not work in case of occlusion, has nothing common with real world

Markerless Tracking

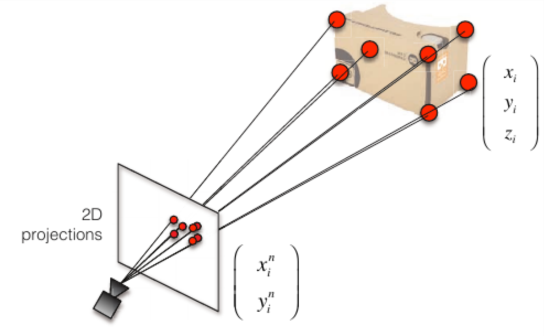
- Markers
  - are cheap and robust against lighting changes
  - but do not work in case of occlusion, has nothing common with real world
- Find natural markers (invariant to scale and rotation)
- Feature extraction → descriptor → feature matching



Scale Invariant Feature Transform (SIFT) [Lowe 1999]

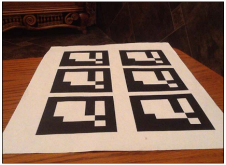
Pose estimation

- how to get project 2D coordinates?
- image formation
- estimate linear homography
- estimate pose from homography

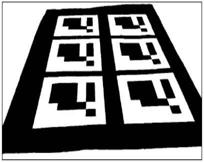


How to get project 2D coordinates

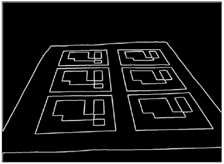
Marker detection



1. Print/Take a picture

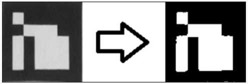


2. Binarise

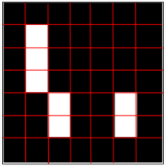


3. Find Contours

4. Warp



5. Threshold (Otsu)



6. Identify

How to get project 2D coordinates

HTC Lighthouse

- Photosensor on the headset
- LEDs and spinning lasers
- Where and when the beam hit the photosensor

`https://www.youtube.com/watch?v=J54dotTt7k0`

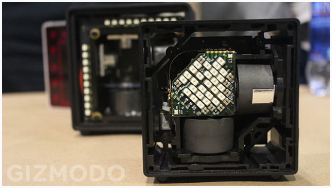


Image formation

a simple model for mapping 3D point coords to 2D

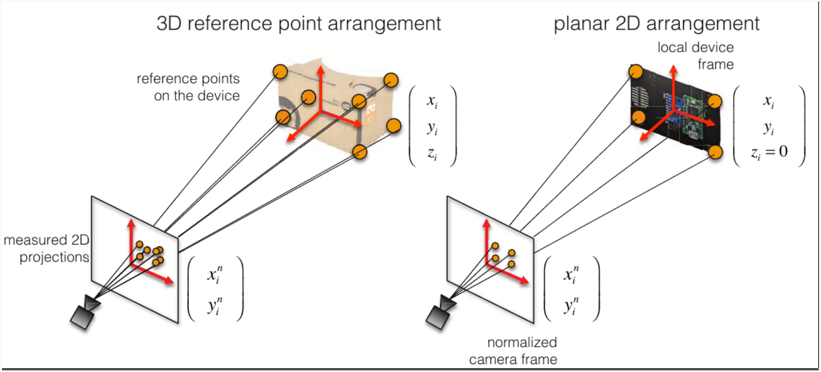
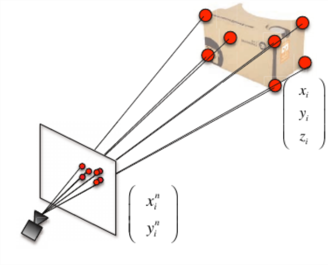


Image formation – 3D arrangement

1. transform 3D point into view space:

$$\begin{pmatrix} x_i^c \\ y_i^c \\ z_i^c \end{pmatrix} = \underbrace{\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{pmatrix}}_{\text{projection matrix}} \underbrace{\begin{pmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{pmatrix}}_{\text{rotation \& translation}} \begin{pmatrix} x_i \\ y_i \\ z_i \\ 1 \end{pmatrix}$$



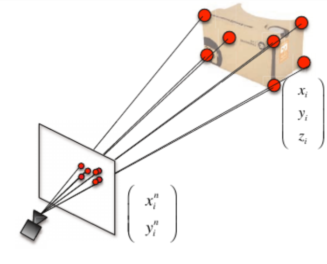
2. perspective divide:

$$\begin{pmatrix} x_i^n \\ y_i^n \end{pmatrix} = \begin{pmatrix} x_i^c / z_i^c \\ y_i^c / z_i^c \end{pmatrix}$$

Image formation – 3D arrangement

1. **transform 3D point into view space:**

$$\begin{pmatrix} x_i^c \\ y_i^c \\ z_i^c \end{pmatrix} = \underbrace{\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{pmatrix}}_{\text{projection matrix}} \underbrace{\begin{pmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{pmatrix}}_{\text{rotation \& translation}} \begin{pmatrix} x_i \\ y_i \\ z_i \\ 1 \end{pmatrix}$$



2. **perspective divide:**

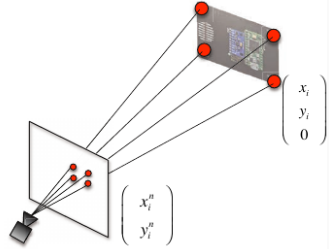
$$\begin{pmatrix} x_i^n \\ y_i^n \end{pmatrix} = \begin{pmatrix} x_i^c / z_i^c \\ y_i^c / z_i^c \end{pmatrix}$$

What happened to my old $p' = M_{vp} M_{proj} M_{cam} M_m p$?

Image formation – 2D arrangement

1. **transform 3D point into view space:**

$$\begin{pmatrix} x_i^c \\ y_i^c \\ z_i^c \end{pmatrix} = \underbrace{\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{pmatrix}}_{\text{projection matrix}} \underbrace{\begin{pmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{pmatrix}}_{\text{rotation \& translation}} \begin{pmatrix} x_i \\ y_i \\ 0 \\ 1 \end{pmatrix}$$



1. **transform 3D point into view space:**

$$\begin{pmatrix} x_i^c \\ y_i^c \\ z_i^c \end{pmatrix} = \underbrace{\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{pmatrix}}_{\text{projection matrix}} \underbrace{\begin{pmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{pmatrix}}_{\text{rotation \& translation}} \begin{pmatrix} x_i \\ y_i \\ 0 \\ 1 \end{pmatrix}$$

$$\begin{pmatrix} x_i^c \\ y_i^c \\ z_i^c \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{pmatrix} \begin{pmatrix} r_{11} & r_{12} & t_x \\ r_{21} & r_{22} & t_y \\ r_{31} & r_{32} & t_z \end{pmatrix} \begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix}$$

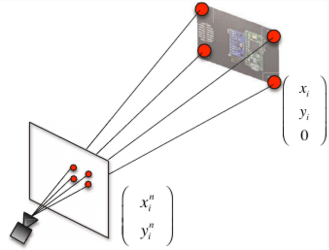
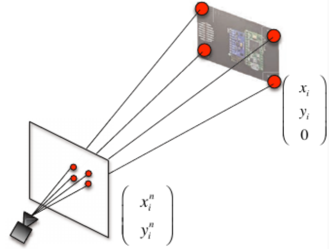


Image formation – 2D arrangement

1. transform 3D point into view space:

$$\begin{pmatrix} x_i^c \\ y_i^c \\ z_i^c \end{pmatrix} = \underbrace{\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{pmatrix}}_{\text{projection matrix}} \underbrace{\begin{pmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{pmatrix}}_{\text{rotation \& translation}} \begin{pmatrix} x_i \\ y_i \\ 0 \\ 1 \end{pmatrix}$$

$$\begin{pmatrix} x_i^c \\ y_i^c \\ z_i^c \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{pmatrix} \begin{pmatrix} r_{11} & r_{12} & t_x \\ r_{21} & r_{22} & t_y \\ r_{31} & r_{32} & t_z \end{pmatrix} \begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix}$$



2. perspective divide:

$$\begin{pmatrix} x_i^n \\ y_i^n \end{pmatrix} = \begin{pmatrix} x_i^c / z_i^c \\ y_i^c / z_i^c \end{pmatrix}$$

The homography matrix

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{pmatrix} \begin{pmatrix} r_{11} & r_{12} & t_x \\ r_{21} & r_{22} & t_y \\ r_{31} & r_{32} & t_z \end{pmatrix} \begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix} = \underbrace{\begin{pmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{pmatrix}}_{\textit{Homography}} \begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix}$$

The homography matrix

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{pmatrix} \begin{pmatrix} r_{11} & r_{12} & t_x \\ r_{21} & r_{22} & t_y \\ r_{31} & r_{32} & t_z \end{pmatrix} \begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix} = \underbrace{\begin{pmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{pmatrix}}_{\textit{Homography}} \begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix}$$

One can scale homography and get the same 3D-to-2D mapping

$$\begin{pmatrix} x_i^n \\ \\ y_i^n \end{pmatrix} = \begin{pmatrix} \frac{x_i^c}{z_i^c} \\ \\ \frac{y_i^n}{z_i^c} \end{pmatrix} = \begin{pmatrix} \frac{sh_1 x_i + sh_2 y_i + sh_3}{sh_7 x_i + sh_8 y_i + sh_8} \\ \\ \frac{sh_4 x_i + sh_5 y_i + sh_6}{sh_7 x_i + sh_8 y_i + sh_8} \end{pmatrix}$$

Computing homography matrix

- estimate a scaled version of homography matrix ($h_9 = 1$)
- we will recover scale factor s later

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{pmatrix} \begin{pmatrix} r_{11} & r_{12} & t_x \\ r_{21} & r_{22} & t_y \\ r_{31} & r_{32} & t_z \end{pmatrix} = s \begin{pmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & 1 \end{pmatrix}$$

Computing homography matrix

$$\begin{pmatrix} x_i^n \\ \\ y_i^n \end{pmatrix} = \begin{pmatrix} \frac{h_1 x_i + h_2 y_i + h_3}{h_7 x_i + h_8 y_i + 1} \\ \\ \frac{h_4 x_i + h_5 y_i + h_6}{h_7 x_i + h_8 y_i + 1} \end{pmatrix}$$

Multiply by denominator

$$x_i^n (h_7 x_i + h_8 y_i + 1) = h_1 x_i + h_2 y_i + h_3$$

$$y_i^n (h_7 x_i + h_8 y_i + 1) = h_4 x_i + h_5 y_i + h_6$$

Computing homography matrix

$$\begin{pmatrix} x_i^n \\ \\ y_i^n \end{pmatrix} = \begin{pmatrix} \frac{h_1 x_i + h_2 y_i + h_3}{h_7 x_i + h_8 y_i + 1} \\ \\ \frac{h_4 x_i + h_5 y_i + h_6}{h_7 x_i + h_8 y_i + 1} \end{pmatrix}$$

Multiply by denominator

$$x_i^n(h_7 x_i + h_8 y_i + 1) = h_1 x_i + h_2 y_i + h_3$$

$$y_i^n(h_7 x_i + h_8 y_i + 1) = h_4 x_i + h_5 y_i + h_6$$

Reorder

$$h_1 x_i + h_2 y_i + h_3 - h_7 x_i x_i^n - h_8 y_i x_i^n = x_i^n$$

$$h_4 x_i + h_5 y_i + h_6 - h_7 x_i y_i^n - h_8 y_i y_i^n = y_i^n$$

Computing homography matrix

- For 8 unknowns, we need 4 3D-2D pairs

$$\underbrace{\begin{pmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1x_1^n & -y_1x_1^n \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -x_1x_1^n & -y_1y_1^n \\ x_2 & y_2 & 1 & 0 & 0 & 0 & -x_2x_2^n & -y_2x_2^n \\ 0 & 0 & 0 & x_2 & y_2 & 1 & -x_2x_2^n & -y_2y_2^n \\ x_3 & y_3 & 1 & 0 & 0 & 0 & -x_3x_3^n & -y_3x_3^n \\ 0 & 0 & 0 & x_3 & y_3 & 1 & -x_3x_3^n & -y_3y_3^n \\ x_4 & y_4 & 1 & 0 & 0 & 0 & -x_4x_4^n & -y_4x_4^n \\ 0 & 0 & 0 & x_4 & y_4 & 1 & -x_4x_4^n & -y_4y_4^n \end{pmatrix}}_{A} \underbrace{\begin{pmatrix} h_1 \\ h_2 \\ h_3 \\ h_4 \\ h_5 \\ h_6 \\ h_7 \\ h_8 \end{pmatrix}}_{h} = \underbrace{\begin{pmatrix} x_1^n \\ y_1^n \\ x_2^n \\ y_2^n \\ x_3^n \\ y_3^n \\ x_4^n \\ y_4^n \end{pmatrix}}_{b}$$

- Solve $Ah = b$

Pose estimation from homography matrix

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{pmatrix} \begin{pmatrix} r_{11} & r_{12} & t_x \\ r_{21} & r_{22} & t_y \\ r_{31} & r_{32} & t_z \end{pmatrix} = s \begin{pmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & 1 \end{pmatrix}$$

this gives us

$$\boxed{t_x = sh_3, \quad t_y = sh_6, \quad t_z = -s.}$$

Pose estimation from homography matrix

Remember

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{pmatrix} \begin{pmatrix} r_{11} & r_{12} & t_x \\ r_{21} & r_{22} & t_y \\ r_{31} & r_{32} & t_z \end{pmatrix} = s \begin{pmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & 1 \end{pmatrix}$$

Rotation matrices are orthonormal

- $\sqrt{r_{11}^2 + r_{21}^2 + r_{31}^2} = 1$ and $\sqrt{r_{12}^2 + r_{22}^2 + r_{32}^2} = 1$
- normalize homography

$$s = \frac{2}{\sqrt{h_1^2 + h_4^2 + h_7^2} + \sqrt{h_2^2 + h_5^2 + h_8^2}}$$

Pose estimation from homography matrix

Remember

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{pmatrix} \begin{pmatrix} r_{11} & r_{12} & t_x \\ r_{21} & r_{22} & t_y \\ r_{31} & r_{32} & t_z \end{pmatrix} = s \begin{pmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & 1 \end{pmatrix}$$

1. Normalize first col of rotation mat

$$\tilde{r}_1 = \begin{pmatrix} h_1 \\ h_4 \\ -h_7 \end{pmatrix}, \qquad r_1 = \frac{\tilde{r}_1}{||\tilde{r}_1||_2}$$

Pose estimation from homography matrix

Remember

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{pmatrix} \begin{pmatrix} r_{11} & r_{12} & t_x \\ r_{21} & r_{22} & t_y \\ r_{31} & r_{32} & t_z \end{pmatrix} = s \begin{pmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & 1 \end{pmatrix}$$

Pose estimation from homography matrix

Remember

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{pmatrix} \begin{pmatrix} r_{11} & r_{12} & t_x \\ r_{21} & r_{22} & t_y \\ r_{31} & r_{32} & t_z \end{pmatrix} = s \begin{pmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & 1 \end{pmatrix}$$

2. Normalize second col after orthogonalization

$$\tilde{r}_2 = \begin{pmatrix} h_2 \\ h_5 \\ -h_8 \end{pmatrix}, \qquad r_2 = \frac{r_1 \times \tilde{r}_2}{||r_1 \times \tilde{r}_2||_2}$$

Pose estimation from homography matrix

Remember

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{pmatrix} \begin{pmatrix} r_{11} & r_{12} & t_x \\ r_{21} & r_{22} & t_y \\ r_{31} & r_{32} & t_z \end{pmatrix} = s \begin{pmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & 1 \end{pmatrix}$$

3. Get third one from cross product

$$r_3 = r_1 \times r_2$$

Pose estimation from homography matrix

Remember Euler angles (yaw-pitch-roll)

$$\underbrace{\begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{pmatrix}}_{\mathbf{R}} = \underbrace{\begin{pmatrix} \cos(\theta_z) & -\sin(\theta_z) & 0 \\ \sin(\theta_z) & \cos(\theta_z) & 0 \\ 0 & 0 & 1 \end{pmatrix}}_{\mathbf{R}_z(\theta_z)} \underbrace{\begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta_x) & -\sin(\theta_x) \\ 0 & \sin(\theta_x) & \cos(\theta_x) \end{pmatrix}}_{\mathbf{R}_x(\theta_x)} \underbrace{\begin{pmatrix} \cos(\theta_y) & 0 & \sin(\theta_y) \\ 0 & 1 & 0 \\ -\sin(\theta_y) & 0 & \cos(\theta_y) \end{pmatrix}}_{\mathbf{R}_y(\theta_y)}$$

$$= \begin{pmatrix} \cos(\theta_y)\cos(\theta_z) - \sin(\theta_x)\sin(\theta_y)\sin(\theta_z) & -\cos(\theta_x)\sin(\theta_z) & \sin(\theta_y)\cos(\theta_z) + \sin(\theta_x)\cos(\theta_y)\sin(\theta_z) \\ \cos(\theta_y)\sin(\theta_z) + \sin(\theta_x)\sin(\theta_y)\cos(\theta_z) & \cos(\theta_x)\cos(\theta_z) & \sin(\theta_y)\sin(\theta_z) - \sin(\theta_x)\cos(\theta_y)\cos(\theta_z) \\ -\cos(\theta_x)\sin(\theta_y) & \sin(\theta_x) & \cos(\theta_x)\cos(\theta_y) \end{pmatrix}$$

Pose estimation from homography matrix

Remember Euler angles (yaw-pitch-roll)

$$\underbrace{\begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{pmatrix}}_{\mathbf{R}} = \underbrace{\begin{pmatrix} \cos(\theta_z) & -\sin(\theta_z) & 0 \\ \sin(\theta_z) & \cos(\theta_z) & 0 \\ 0 & 0 & 1 \end{pmatrix}}_{\mathbf{R}_z(\theta_z)} \underbrace{\begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta_x) & -\sin(\theta_x) \\ 0 & \sin(\theta_x) & \cos(\theta_x) \end{pmatrix}}_{\mathbf{R}_x(\theta_x)} \underbrace{\begin{pmatrix} \cos(\theta_y) & 0 & \sin(\theta_y) \\ 0 & 1 & 0 \\ -\sin(\theta_y) & 0 & \cos(\theta_y) \end{pmatrix}}_{\mathbf{R}_y(\theta_y)}$$

$$= \begin{pmatrix} \cos(\theta_y)\cos(\theta_z) - \sin(\theta_x)\sin(\theta_y)\sin(\theta_z) & -\cos(\theta_x)\sin(\theta_z) & \sin(\theta_y)\cos(\theta_z) + \sin(\theta_x)\cos(\theta_y)\sin(\theta_z) \\ \cos(\theta_y)\sin(\theta_z) + \sin(\theta_x)\sin(\theta_y)\cos(\theta_z) & \cos(\theta_x)\cos(\theta_z) & \sin(\theta_y)\sin(\theta_z) - \sin(\theta_x)\cos(\theta_y)\cos(\theta_z) \\ -\cos(\theta_x)\sin(\theta_y) & \sin(\theta_x) & \cos(\theta_x)\cos(\theta_y) \end{pmatrix}$$

Finally angles

$$r_{32} = \sin(\theta_x) \qquad \Rightarrow \theta_x = \sin^{-1}(r_{32}) = \operatorname{asin}(r_{32})$$

$$\frac{r_{31}}{r_{33}} = -\frac{\cos(\theta_x)\sin(\theta_y)}{\cos(\theta_x)\cos(\theta_y)} = -\tan(\theta_y) \qquad \Rightarrow \theta_y = \tan^{-1}\left(-\frac{r_{31}}{r_{33}}\right) = \operatorname{atan2}(-r_{31}, r_{33})$$

$$\frac{r_{12}}{r_{22}} = -\frac{\cos(\theta_x)\sin(\theta_z)}{\cos(\theta_x)\cos(\theta_z)} = -\tan(\theta_z) \qquad \Rightarrow \theta_z = \tan^{-1}\left(-\frac{r_{12}}{r_{22}}\right) = \operatorname{atan2}(-r_{12}, r_{22})$$

Improve prediction

Predicted $\theta_x, \theta_y, \theta_z, t_x, t_y, t_z$

- Prediction is very sensitive to 2D coordinates

Improve prediction

Predicted $\theta_x, \theta_y, \theta_z, t_x, t_y, t_z$

- Prediction is very sensitive to 2D coordinates
- Use more points than 4

Improve prediction

Predicted $\theta_x, \theta_y, \theta_z, t_x, t_y, t_z$

- Prediction is very sensitive to 2D coordinates
- Use more points than 4
- Apply a simple temporal filter with $0 < a < 1$

$$(\theta_x, \theta_y, \theta_z, t_x, t_y, t_z)_f^{(k)} = \alpha(\theta_x, \theta_y, \theta_z, t_x, t_y, t_z)_f^{(k-1)} + (1-\alpha)(\theta_x, \theta_y, \theta_z, t_x, t_y, t_z)^{(k)}$$

Improve prediction

Predicted $\theta_x, \theta_y, \theta_z, t_x, t_y, t_z$

- Prediction is very sensitive to 2D coordinates
- Use more points than 4
- Apply a simple temporal filter with $0 < a < 1$

$$(\theta_x, \theta_y, \theta_z, t_x, t_y, t_z)_f^{(k)} = \alpha(\theta_x, \theta_y, \theta_z, t_x, t_y, t_z)_f^{(k-1)} + (1-\alpha)(\theta_x, \theta_y, \theta_z, t_x, t_y, t_z)^{(k)}$$

- Combine with inertial sensor predictions (β), typically done with (extended) Kalman filters

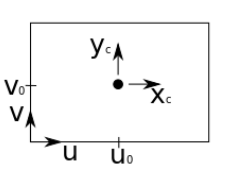
Coordinates in camera frame

$$\begin{pmatrix} x_i^c \\ y_i^c \\ z_i^c \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{pmatrix} \begin{pmatrix} r_{11} & r_{12} & t_x \\ r_{21} & r_{22} & t_y \\ r_{31} & r_{32} & t_z \end{pmatrix} \begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix}$$

Coordinates in image space (pixels)

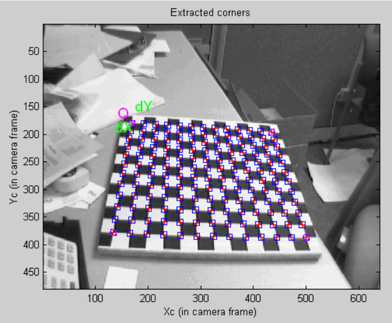
$$\begin{pmatrix} x_i^p \\ y_i^p \end{pmatrix} \approx \underbrace{\begin{pmatrix} f_x & s & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{pmatrix}}_{K} \underbrace{\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{pmatrix} \begin{pmatrix} r_{11} & r_{12} & t_x \\ r_{21} & r_{22} & t_y \\ r_{31} & r_{32} & t_z \end{pmatrix}}_{H} \begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix}$$

- f_x, f_y: focal length
- u_0, v_0: principal point
- s: skew



Camera Calibration

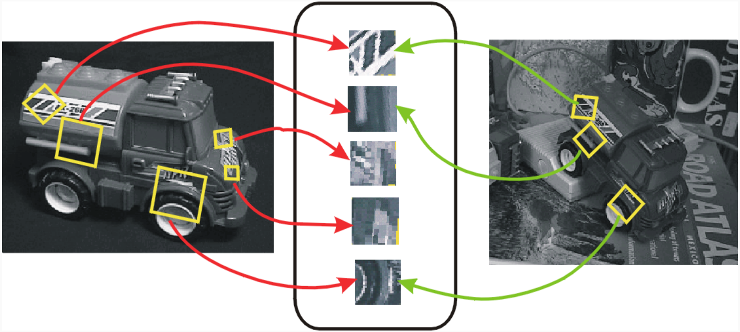
1. print a calibration pattern
2. take pictures
3. extract corners
4. solve $x^p = Px$ where $P = KH$
5. compute intrinsic K and extrinsic parameters H



Markerless tracking

Automatic feature extraction

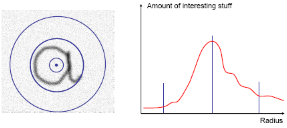
- Keypoint detection (search for locations that are likely to match)
- Descriptor (describe each region around detected keypoint)
- Descriptor Matching (efficiently search for likely matching candidates)

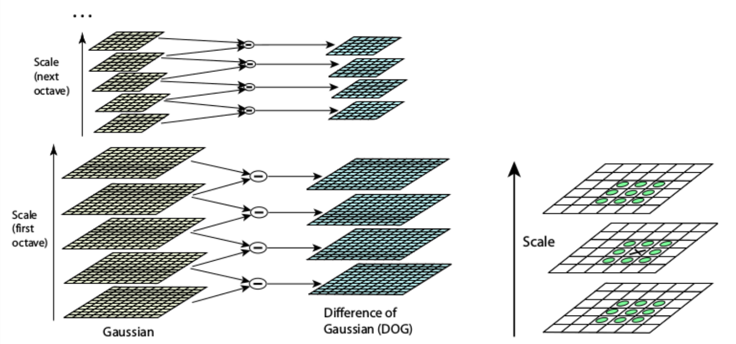


Scale invariant feature transform (SIFT)

1. Scale-space extrema detection [T. Lindeberg 1998]

- Find the points, whose surrounding patches (with some scale) are distinctive





Scale invariant feature transform (SIFT)

2. Orientation Assignment

- Assign an orientation to each keypoint to achieve invariance to rotation
- Compute magnitude and orientation on the Gaussian smoothed images

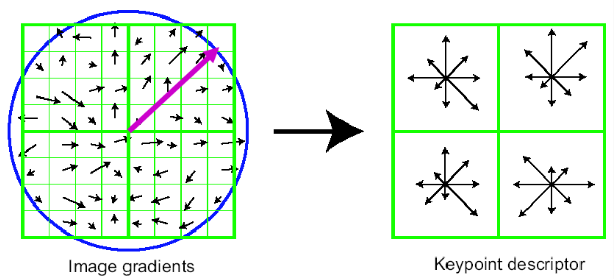
$$m(x,y) = \sqrt{(L(x+1,y) - L(x-1,y))^2 + (L(x,y+1) - L(x,y-1))^2}$$

$$\theta(x,y) = atan(L(x+1,y) - L(x-1,y))/(L(x,y+1) - L(x,y-1))$$

Scale invariant feature transform (SIFT)

3. Descriptors

- We have location, scale and orientation for each keypoint
- Rotate and scale the local region around each keypoint
- Compute a descriptor (4×4 8 bin histograms)



Scale invariant feature transform (SIFT)

4. Matching – Nearest Neighbour

Template

Target

image credit: Hao Jiang.

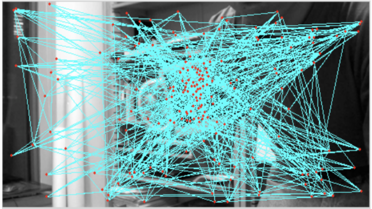
Scale invariant feature transform (SIFT)

4. Matching – Nearest Neighbour

Template

Target

Template



Target

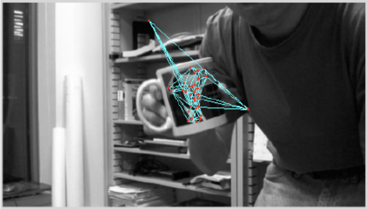
image credit: Hao Jiang.

Scale invariant feature transform (SIFT)

4. Matching – Ratio Test

- x_1 is the first nearest neighbour to x
- x_2 is the second nearest neighbour to x
- $distance(x, x_1')/distance(x, x_2') < 0.7$

Template

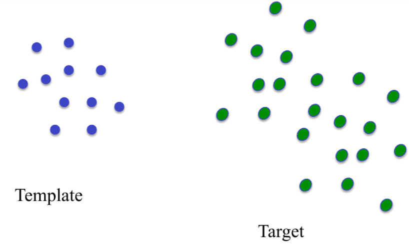


Target

Scale invariant feature transform (SIFT)

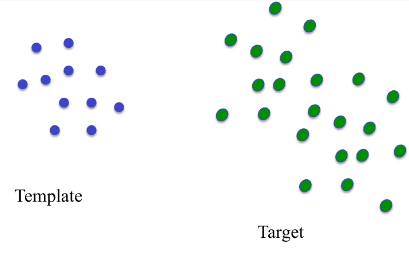
4. **Matching – RANSAC**

   a. Pick three random samples

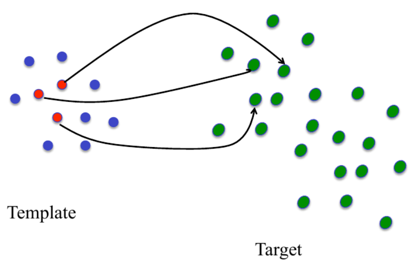


4. **Matching – RANSAC**

a. Pick three random samples



b. Find nearest neighbours



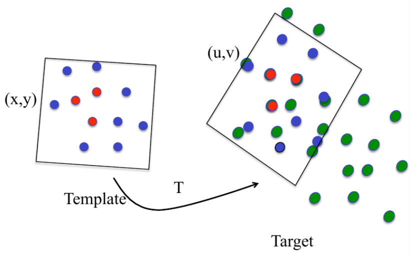
4. Matching – RANSAC

c. Iterate k times

1. Fit a model to inliers

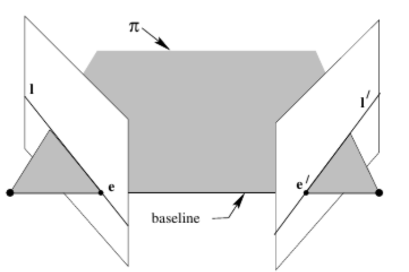
$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{pmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

2. Calculate inliers and outliers



Stereo Vision

- *epipole* is the point of intersection of the line joining the camera centres with the image plane
- *epipolar plane* is a plane containing the baseline
- *epipolar line* is the intersection of an epipolar plane with the image plane
- $x'^T F x = 0$ where F is fundamental matrix



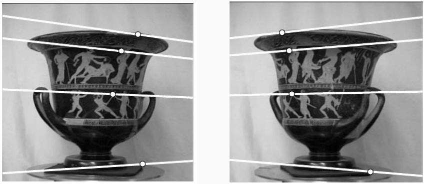
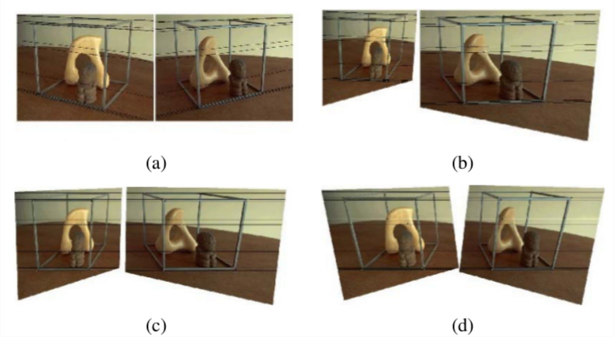


image credit: Hartley & Zisserman.

Stereo Vision – Rectification



a. original image pair overlaid with several epipolar lines
b. images transformed so that epipolar lines are parallel
c. images rectified so that epipolar lines are horizontal and in vertical correspondence
d. final rectification that minimizes horizontal distortions

image credit: Loop & Zhang.

Course material

- Read Chapter 6, Szelisky (http://szeliski.org/Book/)
- Read Chapter 6, 9, LaValle (http://vr.cs.uiuc.edu/)

The End