Introduction to 3D image processing

Topics:

1. Camera model and 3D geometry
2. Depth estimation
3. Point cloud
4. SLAM (simultaneous localization and representation model)
5. NERF model

Overview of 3D image processing

Main object: RGB-D Images

- RGB - 3 color channels
- D - depth, distance from the source to the object in a pixel

Image with the depth also called depths map

Auxiliary object: point clouds

Unordered set of 3D space points. Every point associated with:

- XYZ coordinates
- Optionally: color, class

Represents a single object in a scene

Point clouds tasks

- Classification
- Part segmentation
- Completion
- Semantic segmentation

Depth estimation

Input

- Monocular image / video
- Stereo pairs
- Sparse depth map

Output

- Dense depth map

Simultaneous localization and mapping

SLAM builds a map of an environment and localize the sensor in the map in real time

New View Generation

Task: for a given single view or multiple views with known camera parameters to generate a new view defined by camera's position and direction

Approaches:

- 3D modeling
- Neural scene fitting

Sources of 3D data

(1) Hardware approach: RGB-D sensors

Popular depth sensors
1. Radar
2. Depth camera
3. LIDAR

Radar uses audio waves to determine distances to various objects.
Depending on frequency the range changes.

Advantages:
+ Relatively robust to illumination, weather
+ Cheap

Disadvantages:
- Sparse
- No visual information
- Hard to properly detect small or stationary objects

Depth camera
A setup which estimates depths

Advantages:
+ Sees colors due to RGB camera involved

Disadvantages:
- Sparse depth
- Some cameras have small range
- Heavily affected by the illumination and its intensity

LIDAR

Measures the amount of time of travel to the reflected light from a laser

Adantages:
+ High accuracy for low distance objects

Disadvantages:
- High cost
- "Sees" shapes, not colors
- Affected by weather

(2) Use computer vision to obtain RGB-D image

- Stereo image
- Single image
- Video

(3) Computer vision with hardware

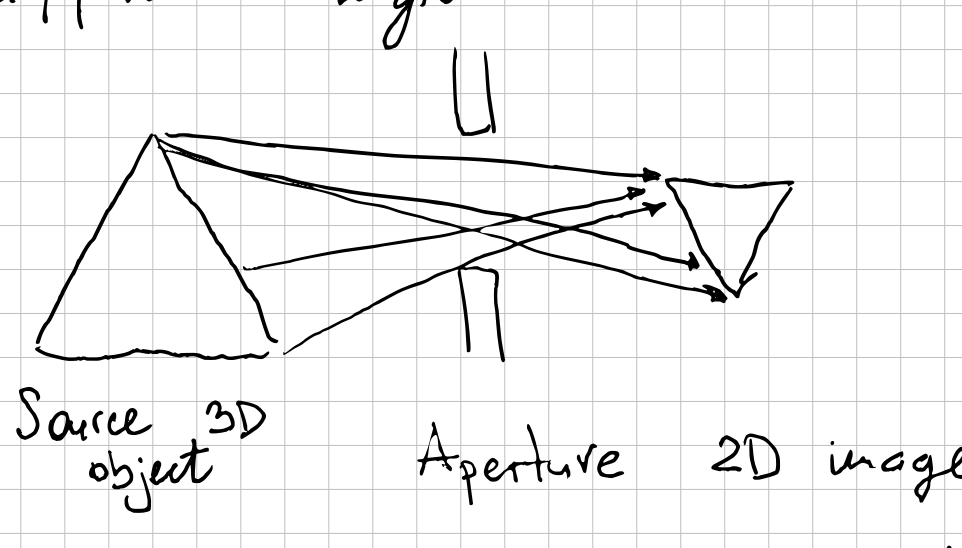
- Turning sparse map into dense map
- Improving map resolution
- "Filling" holes
- Reducing noise
- Combine color and depth info to improve analysis

Basics of 3D image processing

Camera model

Perception

An object emits light rays which are registered by some sensor. Each point of interest generates various rays at different angles.



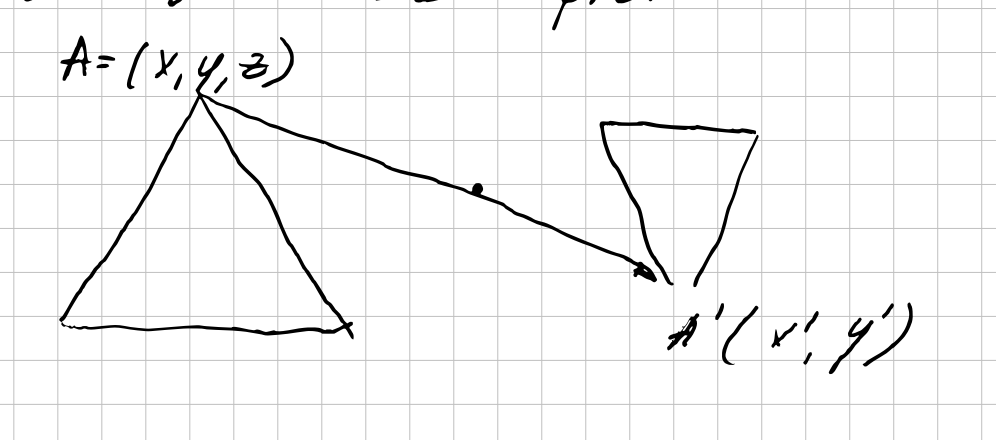
Each image point gets affected by rays from multiple points of the source object resulting in blurry image.

Solution: limit aperture width, so it does not let so many rays into sensor.

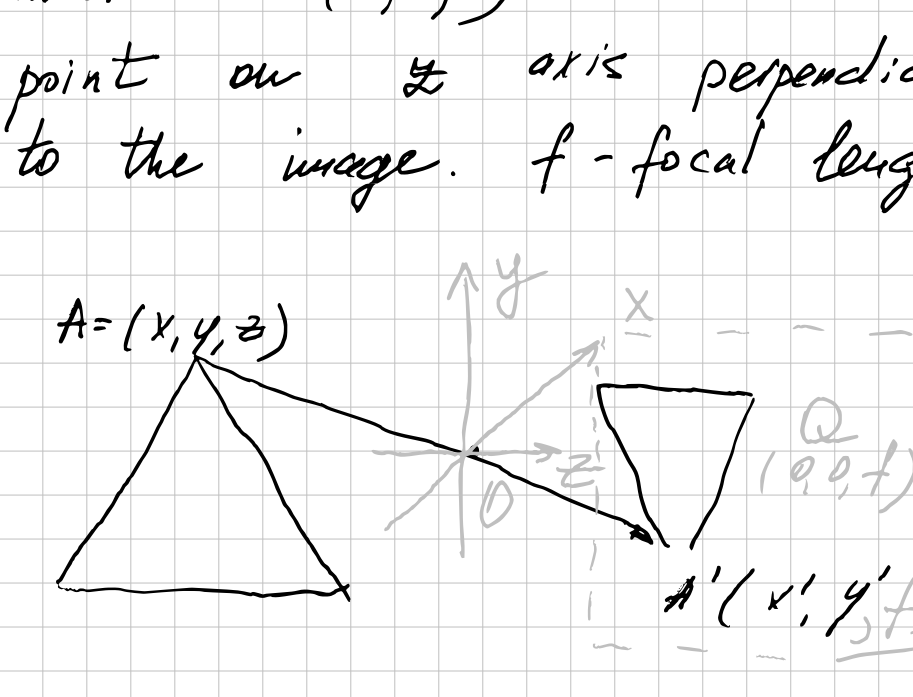
Assuming continuous space and image there will be more than one ray affecting infinitely many points of the image.

Solution: make aperture a single point with width 0.

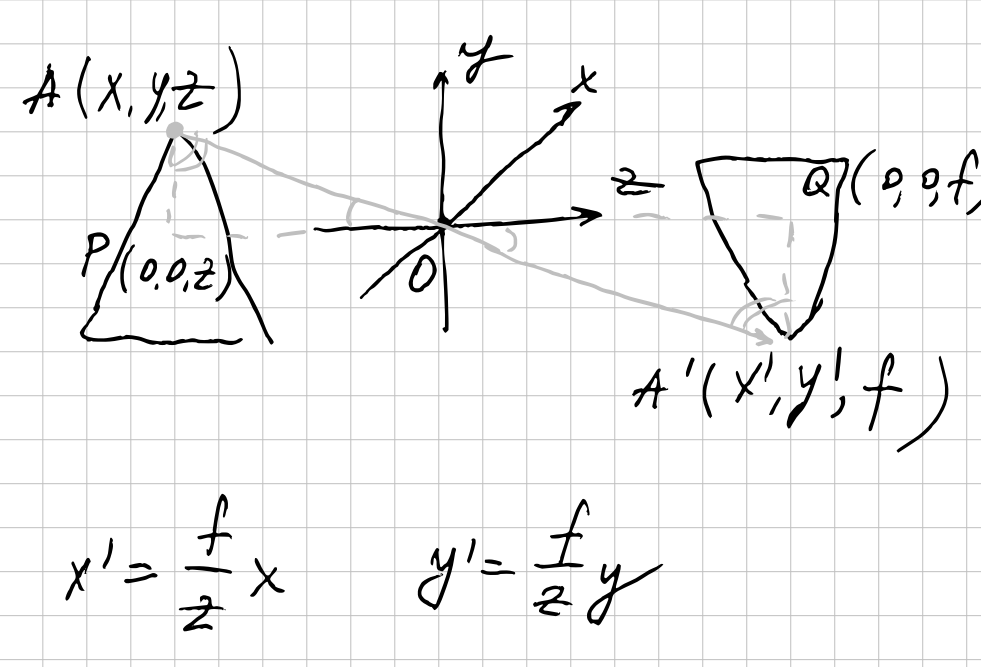
Pinhole camera model



There is a point $A(x, y, z)$ on the real 3D object. How to get its position $A'(x', y')$ on the 2D photo?

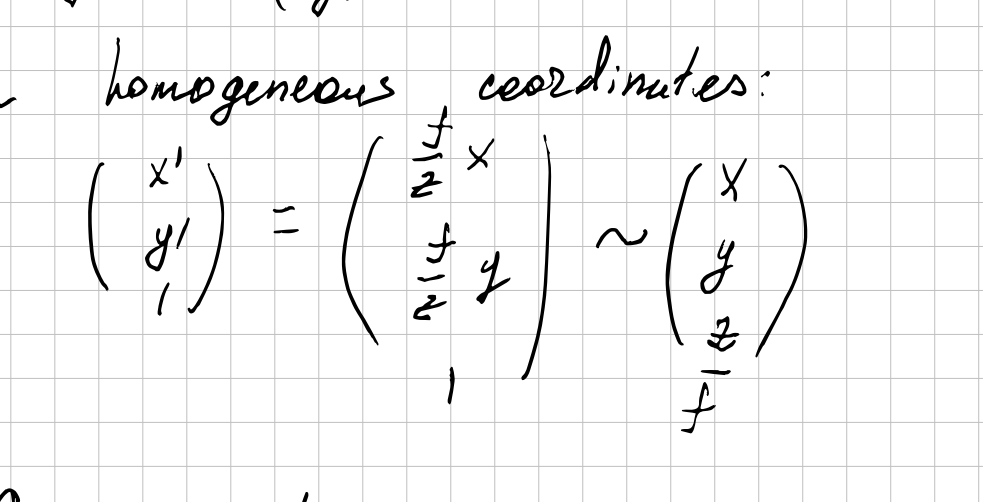


Introduce a coordinate system on a pinhole camera $O(0,0,0)$. Q is a point on z axis perpendicular to the image. f - focal lenght



By symmetry around O: $x' = -C \cdot x$; $y' = -C \cdot y$

By similarity $APO \sim A'QO$: $\frac{QO}{PO} = \frac{f}{z}$



$$x' = \frac{f}{z} x \qquad y' = \frac{f}{z} y$$

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \frac{f}{z} \begin{pmatrix} x \\ y \end{pmatrix}$$

In homogeneous coordinates:

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} \frac{f}{z} x \\ \frac{f}{z} y \\ 1 \end{pmatrix} \sim \begin{pmatrix} x \\ y \\ \frac{z}{f} \end{pmatrix}$$

Camera matrix

As a linear transformation:

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} \sim \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{1}{f} & 0 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

Camera matrix:

$$C = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{1}{f} & 0 \end{pmatrix} \sim \begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

A canonical camera is a camera with focal length of 1 unit: $f = 1$. It has a camera matrix:

$$C = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} = (I | 0)$$

Canonical cameras are more convenient for calculations.

Pinhole Camera Model

Positive features:

+ Provides with a convenient 3D to 2D mapping.

Disadvantages:

- Impossible to build in real life
- limits amount of light which leads to very dim images

Long Exposure Time

Possible solution: longer exposure time for more light coming through the pinhole.

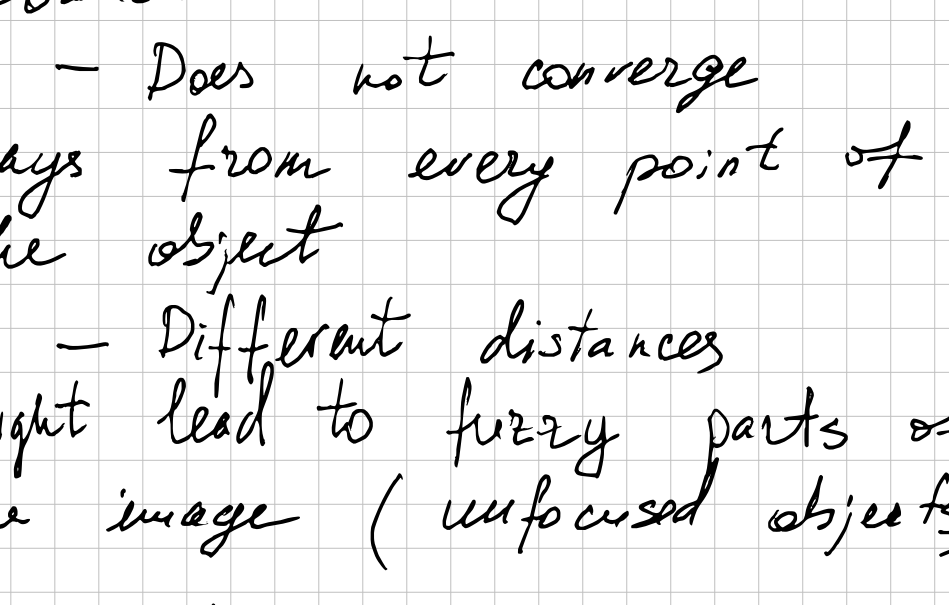
Core problem: blurry images

Sources of such blur:

1. Moving objects (in different time they emit light from different relative positions)
2. Not stabilised/shaking camera

Lenses

Converge all light rays into the same point.



Upsides:

+ Solves the problem of pinhole camera

Downsides:

- Does not converge rays from every point of the object
- Different distances might lead to fuzzy parts of the image (unfocused objects)
- Various geometric distortions
- Might introduce vignetting

Camera physical matrix

Usually, each single pixel of the camera matrix registers only one of the colors a raw image has only R, G, or B in every pixel. The layout of R, G, B filters is called a color filter mosaic.

The Bayer color filter (CFM) is one of the most popular layouts.

The Bayer CFM

In the Bayer CFM for each square 2x2 pixels, there are 2 pixels of G, one R, one B. A camera matrix then consists of such repeated 2x2 squares with the same pattern.

G R
B G

Demosaicing

To recreate a full colored image demosaicing algorithms are used. Simplest methods either interpolate colors from a set of close neighbours or use a whole 2x2 square to describe a single fully colored pixel (reduces image width & height)

One 2x2 patch of the shape (2,2,1) → One colored pixel of the shape (1,1,3)

G R
B G

→

R, $\frac{G+G}{2}$, B

Often some additional steps like gamma correction or automatic white balance might be applied by the camera before outputing a final image.

Depth Estimation

Depth estimation is the task of the estimating distance relative to camera for all objects and surfaces.

Applications:

- 3D scene reconstruction
- self-driving cars
- 3D object reconstruction
- Augmented Reality

Datasets:

- KITTI
- Cityscapes
- SYNTHIA
- Middlebury
- NYUv2

Metrics:

- Mean absolute relative error (abs_rel)

$$abs\_rel = \frac{1}{N}\sum_{i=1}^{N}\frac{\|\hat{d}_i - d_i\|}{d_i}$$

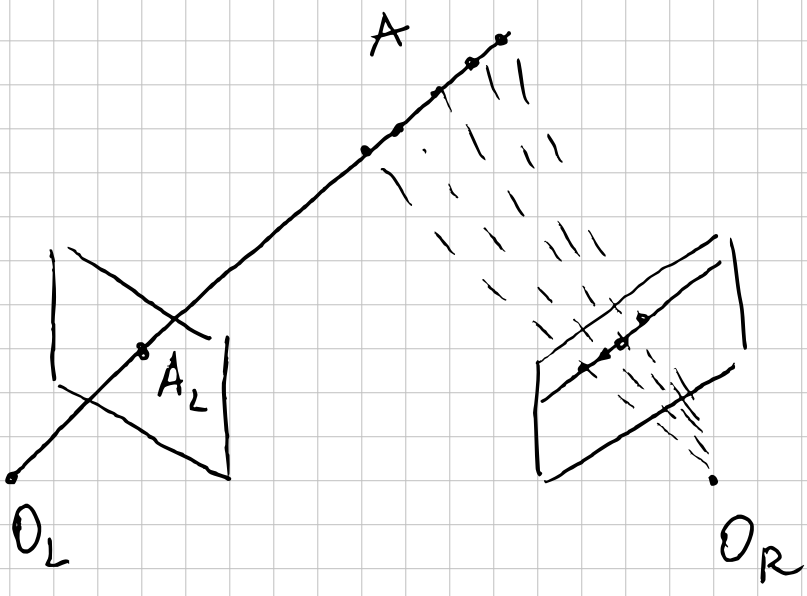
- Squared relative error

$$sq\_rel = \frac{1}{N}\sum_{i=1}^{N}\frac{\|\hat{d}_i - d_i\|^2}{d_i}$$

- Root Mean Square Error

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\|\hat{d}_i - d_i\|^2}$$

Stereo-based depth estimation

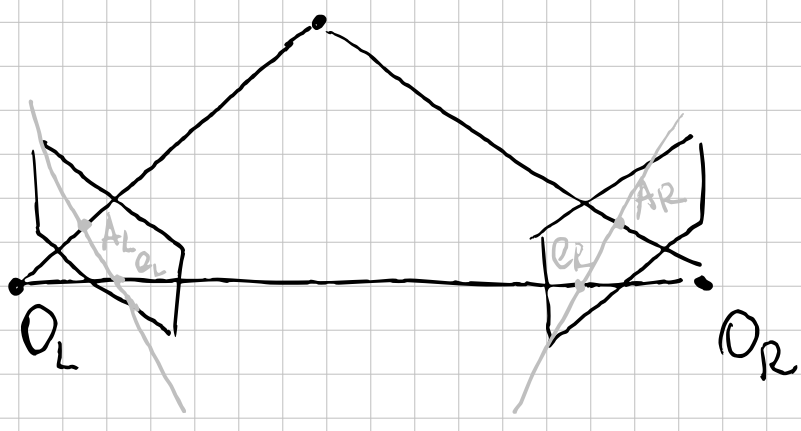


A - point on a 3D object

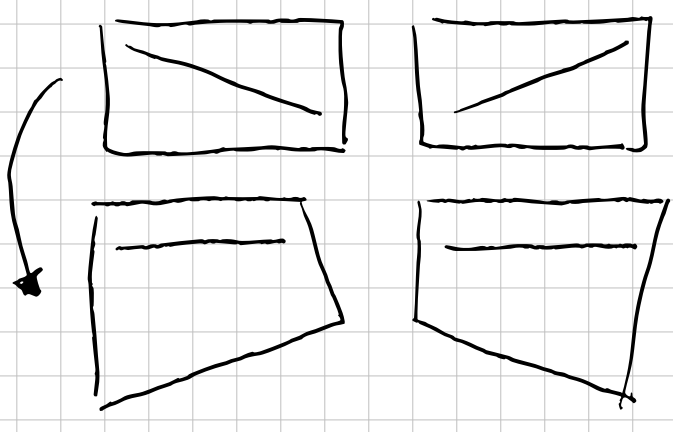
O_L, O_R - locations of left & right cameras

A_L - projection of A on left image

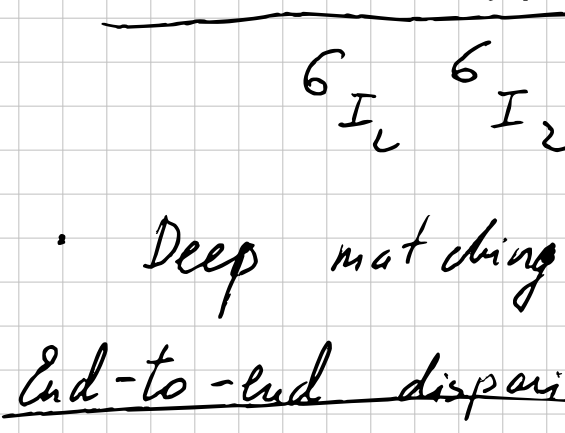
- Can find correspondence for A_L on O_R
- Can estimate 3D location of point A



Stereo rectification is a process of mating parallel lines horizontal



Depth from stereo



$$\frac{T}{Z} = \frac{T + x_L - x_r}{Z - f}$$

$$Z = \frac{f \cdot T}{\underbrace{x_r - x_L}_{\text{disparity}}}$$

How can we estimate disparity?

- All possible matching points on epipolar line
- Patches can be compared by computing matching cost

Matching cost

- Squared difference $(I_r - I_l)^2$
- Normalised Cross correlation

$$\frac{E[(I_l - \mu_{I_l})(I_2 - \mu_{I_2})]}{\sigma_{I_l}\,\sigma_{I_2}}$$

- Deep matching cost

End-to-end disparity regression

- Learn deep representation

Monocular depth estimation

- Supervised
  - DORN
    - Introduce spacing-increasing discretization strategy
    - Remove subsampling in the last pooling layers
    - ASPP module consists of dilated convolutions
  - MIDAS
    - Use depth dataset obtained from 3D movies
    - Introduce scale and bias invariant loss
    - Train model on multiple dataset
    - Test on unseen data

Drawbacks

- Best models requires a lot of ground truth depth maps
- Hard to obtain high quality depth
- Time synchronization of a camera and depth sensor
- Poor generalization

Depth sources

- RADAR
  Sparse, weather dependent
- LiDAR
  Expensive, weather dependent
- 3D movies
  Only relative depth

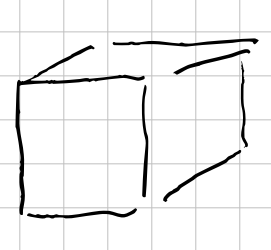
Self-supervised depth estimation

- Train require only stereo-pairs or videos
- Model predicts depth and relative pose (6-DOF)

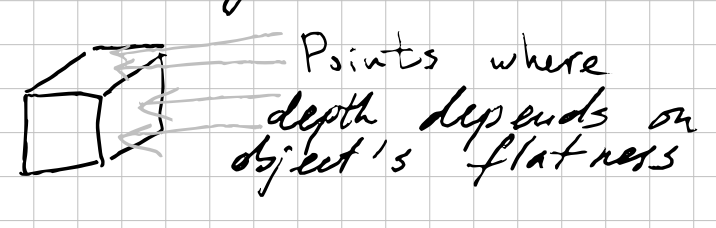
Monodepth 2

Epipolar geometry

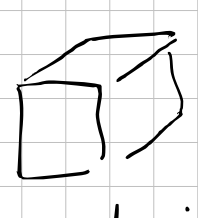
Cube or flat polyhedron?



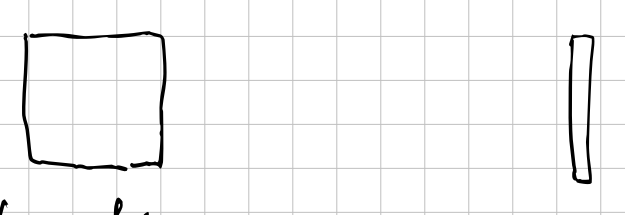
Issue: depth depends on flatness of an object. Information about it is lost due to 3D object → 2D image translation.



Solution: side view



Original image

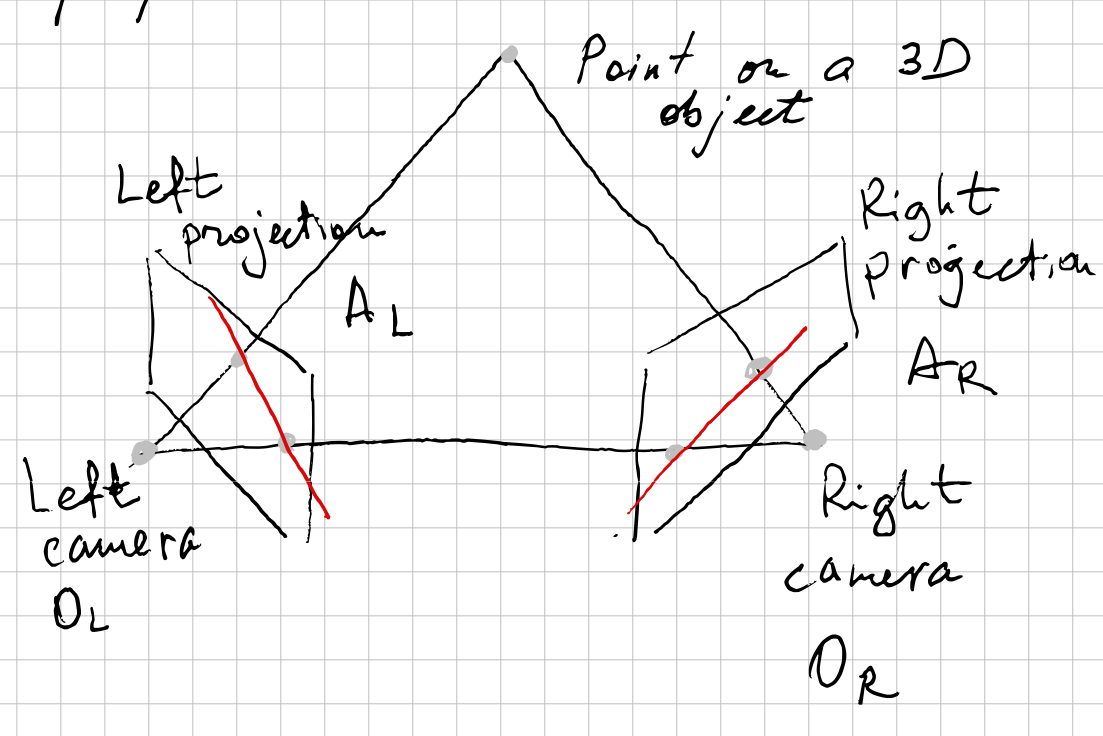


View from right (3D)

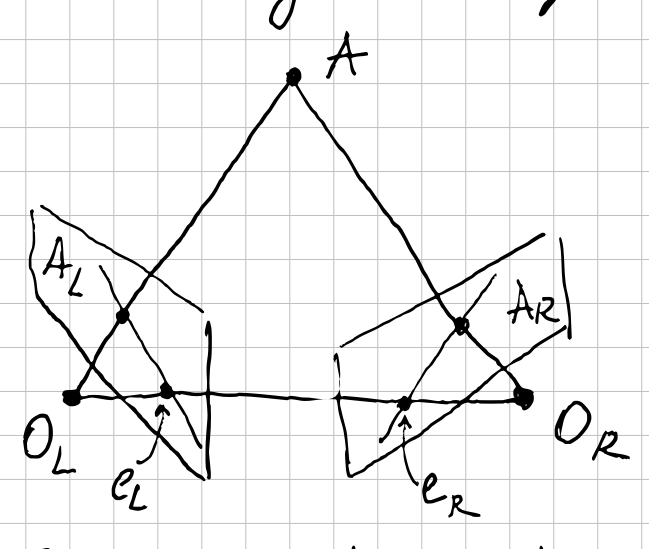
View from right (flat object)

Epipolar geometry

2 cameras view the same 3D object from 2 different positions at the same time.
Result: 2 related 2D projections.



A - point on a 3D object
O_L, O_R - locations of left & right cameras
A_L, A_R - projections of A on left and right images



$O_L A O_R$ - epipolar plane
$O_L O_R$ - base line
e_L, e_R - left and right epipoles (intersection of a baseline and left/right image)

$A_L e_L$, $A_R e_R$ - epipolar lines

Projections on both planes lie on respective epipolar lines. Knowing A_L we know that A_R will be on the right epipolar plane.

Consider coordinate systems related to 2 canonical cameras:

Point A in:
- left coordinate system: (a_1, a_2, a_3)
- right coordinate system: (a_1', a_2', a_3')

Point A_L in left camera's coordinate system:

$$\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \frac{1}{a_3} \begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix} \leftrightarrow \overline{x} = \frac{1}{a_3} \overline{a}$$

Point A_R in right camera's coordinate system:

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \frac{1}{a_3'} \begin{pmatrix} a_1' \\ a_2' \\ a_3' \end{pmatrix} \leftrightarrow \overline{x'} = \frac{1}{a_3'} \overline{a'}$$

A_R could be obtained from A_L by rotation and translation. Both are linear transformations.
$\overline{x'} = R \cdot (\overline{x} - t)$, where
R - rotation; t - translation vector.

Essential matrix

$$\overline{x} = \frac{1}{a_3} \overline{a}$$

$$\overline{x'} = \frac{1}{a_3'} \overline{a'}$$

$$\overline{x'} = R(\overline{x} - t)$$

Then $E = R \cdot [t]_x$, where
$[t]_x$ - matrix representation of cross product

$$[t]_x = \begin{bmatrix} 0 & -t_3 & t_2 \\ t_3 & 0 & -t_1 \\ -t_2 & t_1 & 0 \end{bmatrix}$$

Core property $(\overline{x'})^T \cdot E \cdot \overline{x} = 0$

Properties:

$$t^T \cdot [t]_x$$

$$= (t_1, t_2, t_3) \begin{bmatrix} 0 & -t_3 & t_2 \\ t_3 & 0 & -t_1 \\ -t_2 & t_1 & 0 \end{bmatrix}$$

$$= (0t_1 + t_2 t_3 - t_3 t_2, -t_1 t_3 + 0t_2 + t_1 t_3, t_1 t_2 - t_2 t_1 + 0t_3) = \vec{0}$$

$$x^T \cdot [t]_x \cdot x$$

$$= (x_1, x_2, x_3) \begin{bmatrix} 0 & -t_3 & t_2 \\ t_3 & 0 & -t_1 \\ -t_2 & t_1 & 0 \end{bmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \ominus$$

$$\ominus (x_2 t_3 - x_3 t_2, -x_1 t_3 + x_3 t_1, x_1 t_2 - x_2 t_1) \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = (x_1 x_2 t_3 - x_1 x_3 t_2 - x_1 x_2 t_3 + x_2 x_3 t_1 + x_1 x_3 t_2 - x_2 x_3 t_1) = 0$$

Main property of the essential matrix

$$(\overline{x'})^T \cdot E \cdot \overline{x} = 0$$

$$E = R \cdot [t]_x \quad (1)$$
$$\overline{x'} = R \cdot (\overline{x} - t) \quad (2)$$
$$t^T \cdot [t]_x = 0 \quad (3)$$
$$x^T \cdot [t]_x \cdot x = 0 \quad (4)$$
$$R^T \cdot R = I \quad (5)$$

Checking main property

$$(\overline{x'})^T \cdot E \cdot \overline{x} = 0 \quad (\text{Using}(1)(2))$$

$$(\overline{x} - t)^T \cdot R^T \cdot R [t]_x \cdot \overline{x} =$$

$$= (\overline{x} - t)^T [t]_x \overline{x} \ominus (\text{brackets})$$

$$\ominus \overline{x}^T [t]_x \overline{x} - t^T [t]_x \overline{x} =$$

$$= 0 \overline{x} = 0$$

Q.E.D.

If coordinate systems are not with canonical cameras, essential matrix turns into

Fundamental matrix

$$E = (K')^T F K$$

K', K - intrinsic calibration matrices

$$F = (K')^{-T} E K^{-1}$$
$$F = (K')^{-T} \cdot R \cdot [t]_x \cdot K^{-1}$$

Fundamental and essential matrices provide a way to relate a 3D point in real world to its 2 projections (images) made by 2 cameras. This might be used to obtain depth values and coordinates of the 3D object from a stereo image.

Intrinsic calibration matrix

Intrinsic calibration matrix incorporates information about camera model's intrinsic parameters: $a_x = f m_x$, $a_y = f \cdot m_y$ - focal length in pixels, γ - skew coefficient of y and x axes, u_0, v_0 - principal points. Each specific camera model might have its own values.

$$K = \begin{pmatrix} a_x & \gamma & u_0 & 0 \\ 0 & a_y & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

Stereo systems

Stereo Pair of Images is a pair of images of the same scene made by different cameras in different positions.

- Stereo pair is used as an input or additional data (usually boosts model performance)
- Train with stereo pairs, inference with monocular distilled model
- Use sequences of images from a video as pseudo-stereo pair. Camera moves slightly between frames. With appropriate fps: not fast enough to change the scene, but fast enough to get a slightly different image

Localization and mapping

SLAM - simultaneous localization and mapping

localization - finding where the robot is with respect to the map

mapping - building a representation of the environment

Visual SLAM uses only images

Data sources

- Mono camera
- Stereo camera
- Depth sensor
- Inertial measurement unit (IMU)

Mono camera

- Low cost
- Scale ambiguity of a depth
- Depth can be estimated only by neural networks
- High computational cost

Stereo camera

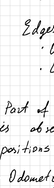
- Accurate absolute depth
- Complicated calibration process
- Very high computational cost

RGB-D camera

- True depth map
- Save computational resources
- Time synchronization with camera
- Noisy / sparse depth
- Issues with measure distance to transparent/reflecting surfaces

IMU

- Measures the angular velocity and the acceleration
- Accelerometer + gyroscope
- Measurements are noisy



Mathematical formulation

Input

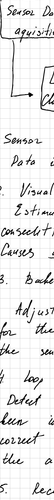
$U_{0:k} = \{u_0, u_1, \ldots, u_k\}$ - input commands

$Z_{0:k} = \{z_0, z_1, \ldots z_k\}$ - sensors measurements

Output

$X_{0:k} = \{x_0, x_1, \ldots, x_k\}$ - estimated positions

$m = \{m_0, m_1, \ldots m_k\}$ - landmarks



- estimated

- ground truth

Graph based SLAM

Nodes:
- Locations (poses)
- Landmarks

Edges:
- Odometry measurement (estimation)
- Landmark observations

- Part of environment (landmark m_j) is observed from x_{k-1}, x_k, x_{k-1} positions
- Odometry between positions can be estimated based on common landmarks

Slam General schema

1. Sensor data
   Data is read from cameras
2. Visual odometry
   Estimate the motion between consecutive camera frames
   Causes accumulative drift
3. Backend optimization
   Adjust visual odometry results for the noise caused by the sensors
4. Loop closing
   Detect when a point has been identified twice, and correct the entire map for the accumulated drift error
5. Reconstruction
   - Build the map

Maps:
- Metric
- Topological

Problems of modern SLAM

- Lighting conditions
- Interacting with people
- Solution for joint indoor and outdoor environments

ORB SLAM

Parallel steps:

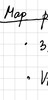
- Tracking
- Local mapping
- Loop closing
- Open source
- Real-time operations in large environments
- ORB feature detector
- Same features for all steps

ORB — oriented FAST and Rotated BRIEF

- FAST (Features from Accelerated Segment Test) is a high-speed corner detector
- BRIEF (Binary Robust Independent Elementary Features) is a feature point descriptor

FAST

- Checks 16 pixels in a neighborhood
- Corner is detected if 12 sequential pixels are brighter or darker than the central pixel
- First checks only 4 pixels: 1, 9, 5, and 13



BRIEF

- Smoothing an image
- Choosing random # of points for each image patch
- Comparison of the pixel intensity of pairs of points
- Result n-dimensional bit string
- n: 128, 256 or 512

Map points

- 3D position
- Viewing direction
- ORB descriptor
- Max & min distances at which the point can be observed

Key frames

- Camera pose
- Camera intrinsics: focal length, principal point
- ORB features

Tracking

1. ORB extraction
2. Initial Pose estimation from previous frame
3. If the tracking is lost: initial pose estimation via Global Relocalization
4. Track local map
5. New Key Frame decision

Local Mapping

Covisibility graph:
- Graph of keyframes
- Edge if keyframes share map points

Essential graph:
- All keyframes
- Contains a spanning tree of edges

1. Keyframe Insertion
2. Recent Map Points Culling
3. New Map Point Creation
4. Local bundle adjustment
5. Local Keyframe Culling

Loop Closing

1. Loop Candidates Detection
2. Compute the Similarity Transformation
3. Loop Fusion
4. Essential Graph Optimization

ORB SLAM-2

- Add stereo and rgb-d cameras
- More accurate
- Can estimate 3D position of a point

ORB SLAM-3

- Integrate IMU
- Improved recall-place recognition

Deep Learning for SLAM

- Visual odometry
- SLAM
- Feature detectors & descriptors

- DeepVO
- SSL VO + Depth
- CNN SLAM (monocular)
- Super Point

Novel view synthesis

Input: set of sparse images representing some object or scene, novel viewpoint position

Output: image representing object or scene from the given viewpoint

Approaches:

1. Modelling (object reconstruction)
2. Image-based rendering (NeRF)

Voxel-based approaches

3D-R2N2

Pix2Vox

TMVNet

Mesh-based approaches.

- Pixel2Mesh
- Textured Mesh Gen
- AtlasNet

Implicit shape representation

- Occupancy Networks