Script **generated by TTT**

Title: Sturm: Visual Navigation (26.06.2012)

Date: Tue Jun 26 10:18:22 CEST 2012

Duration: 89:39 min

Pages: 97

Computer Vision Group
Prof. Daniel Cremers
Technische Universität München

Visual Navigation for Flying Robots

Place Recognition, ICP, and Dense Reconstruction

Dr. Jürgen Sturm

Exercise Sheet 5

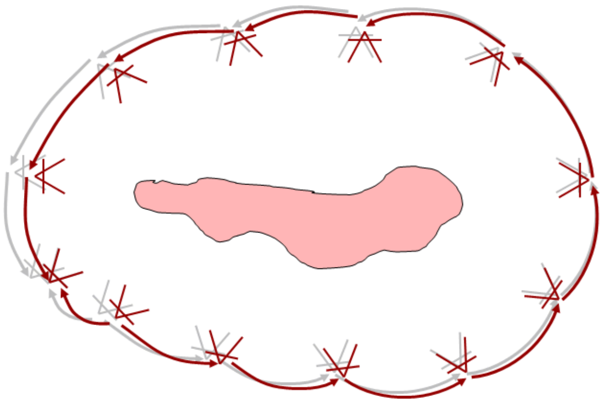
- Prepare mid-term presentation
- Proposed structure: 3 slides
  1. Remind people who you are and what you are doing (can be same slide as last time)
  2. Your work/achievements so far (video is a plus)
  3. Your plans for the next two weeks
- Hand in slides before July 3, 10am

Agenda for Today

- **Localization**
  - Visual place recognition
  - Scan matching and Iterative Closest Point
- Mapping with known poses (3D reconstruction)
  - Occupancy grids
  - Octtrees
  - Signed distance field
  - Meshing

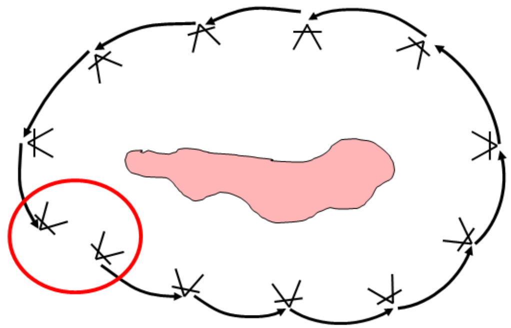
Remember: Loop Closures

- Use loop-closures to minimize the drift / minimize the error over all constraints



Loop Closures

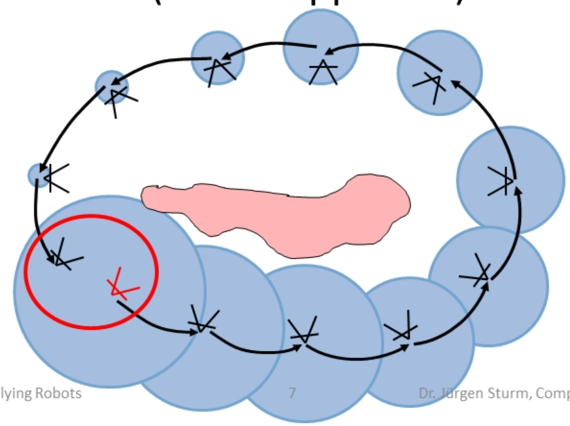
How can we detect loop closures efficiently?



Loop Closures

How can we detect loop closures efficiently?

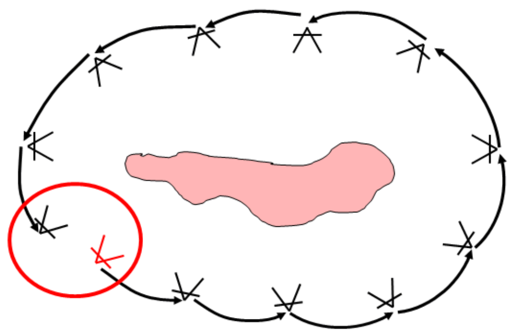
2. Use motion model and covariance to limit search radius (metric approach)



Loop Closures

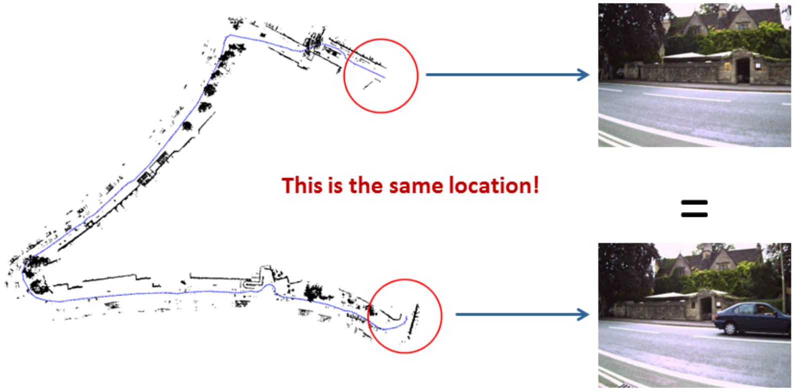
How can we detect loop closures efficiently?

3. Appearance-based place recognition (using bag of words)

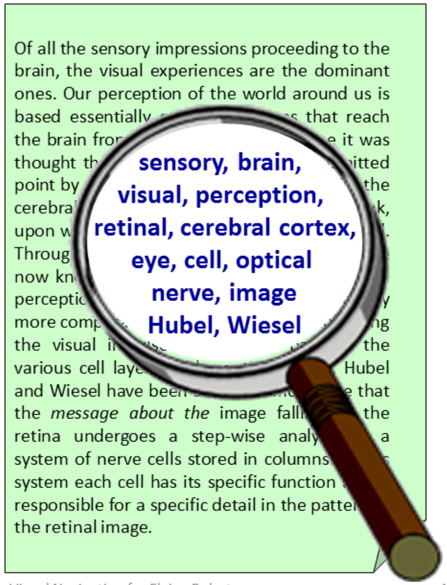


Appearance-based Place Recognition

Appearance can help to recover the pose estimate where metric approaches might fail



Analogy to Document Retrieval



Object/Scene Recognition

- Analogy to documents: The content can be inferred from the frequency of visual words

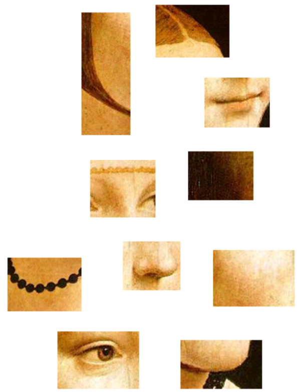
object

bag of visual words

Bag of Visual Words

- Visual words = (independent) features

face



features

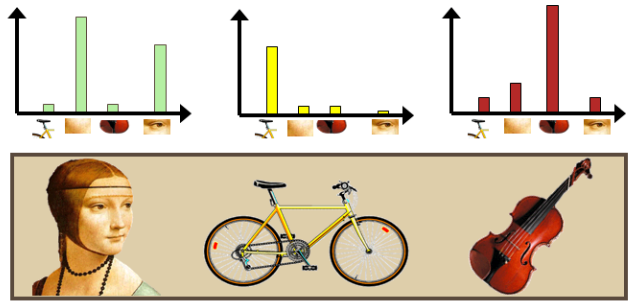
Bag of Visual Words

- Visual words = (independent) features
- Construct a dictionary of representative words

dictionary of visual words (codebook)

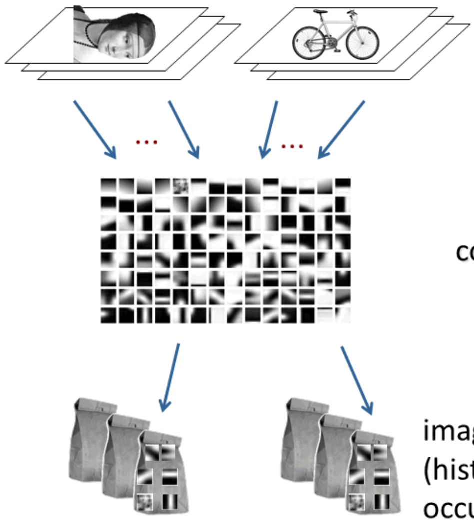
Bag of Visual Words

- Visual words = (independent) features
- Construct a dictionary of representative words
- Represent the image based on a histogram of word occurrences (bag)



Each detected feature is assigned to the closest entry in the codebook

Overview

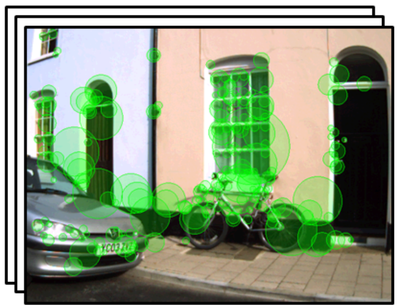


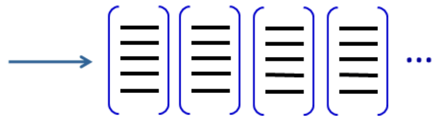
feature detection and extraction (e.g., SIFT, …)

codewords dictionary

image representation (histogram of word occurrences)

Learning the Dictionary

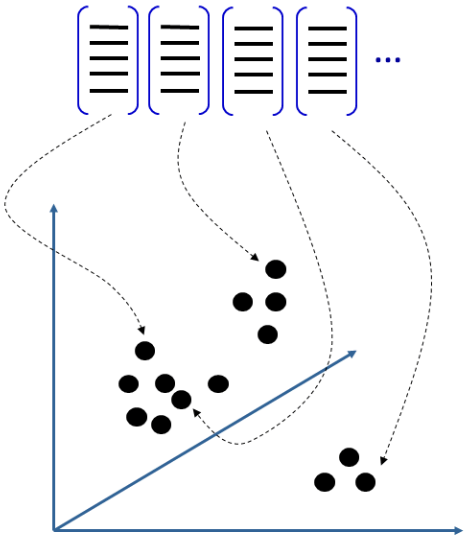




descriptor vectors (e.g., SIFT, SURF, …)

example patch

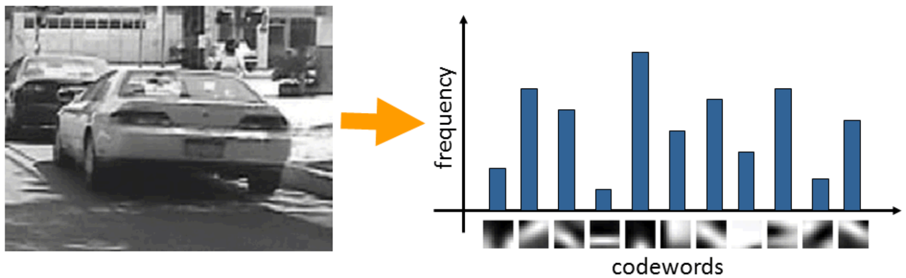
Learning the Dictionary



Learning the Visual Vocabulary

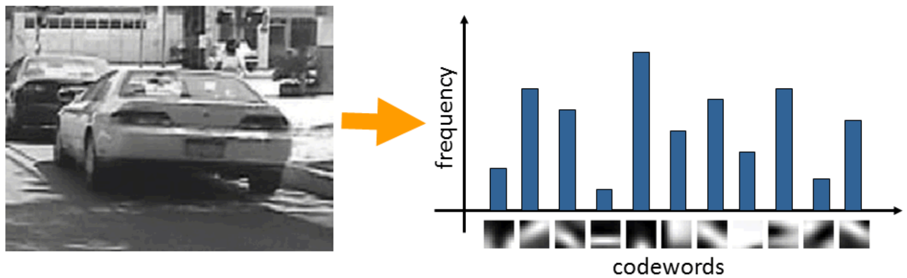
Example Image Representation

- Build the histogram by assigning each detected feature to the closest entry in the codebook



Example Image Representation

- Build the histogram by assigning each detected feature to the closest entry in the codebook

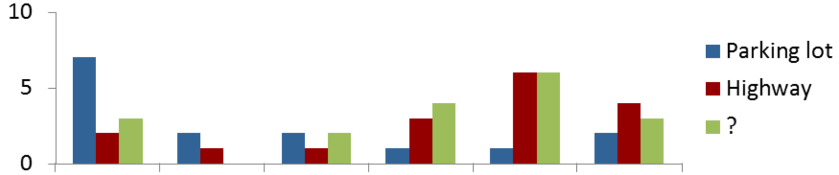


Object/Scene Recognition

- Compare histogram of new scene with those of known scenes, e.g., using
  - simple histogram intersection

$$score(\mathbf{p}, \mathbf{q}) = \sum \min(p_i, q_i)$$

  - naïve Bayes
  - more advanced statistical methods

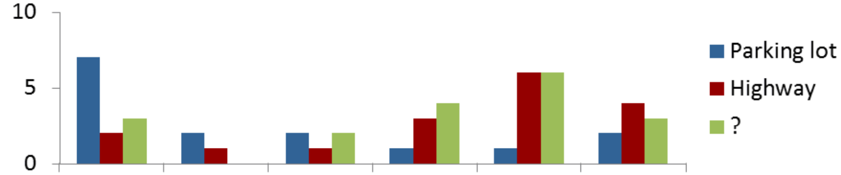


Object/Scene Recognition

- Compare histogram of new scene with those of known scenes, e.g., using
  - simple histogram intersection

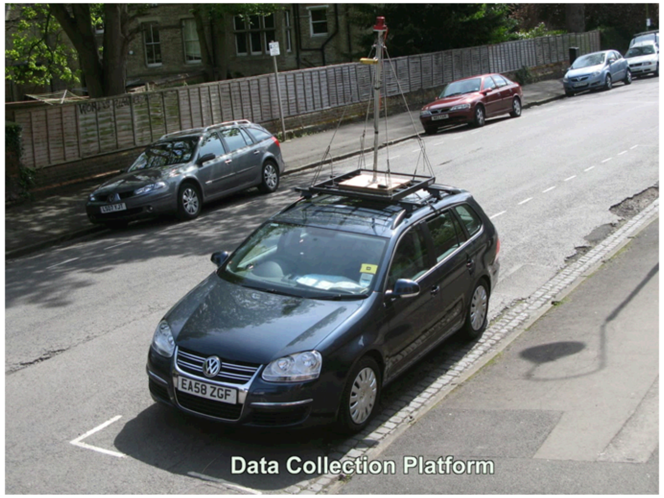
$$score(\mathbf{p}, \mathbf{q}) = \sum \min(p_i, q_i)$$

  - naïve Bayes
  - more advanced statistical methods



Example: FAB-MAP

[Cummins and Newman, 2008]



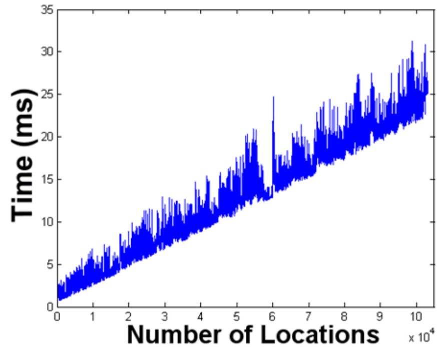
Example: FAB-MAP

[Cummins and Newman, 2008]

Example: FAB-MAP
[Cummins and Newman, 2008]

Timing Performance

- Inference: 25 ms for 100k locations
- SURF detection + quantization: 483 ms

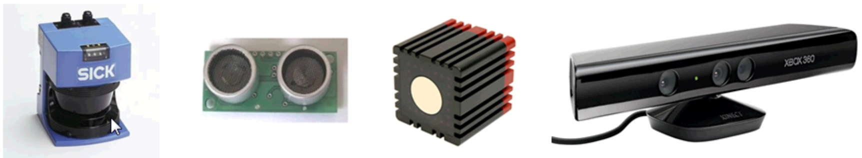


Summary: Bag of Words
[Fei-Fei and Perona, 2005; Nister and Stewenius, 2006]

- Compact representation of content
- Highly efficient and scalable
- Requires training of a dictionary
- Insensitive to viewpoint changes/image deformations (inherited from feature descriptor)

Laser-based Motion Estimation

- So far, we looked at motion estimation (and place recognition) from **visual** sensors
- Today, we cover motion estimation from **range** sensors
  - Laser scanner (laser range finder, ultrasound)
  - Depth cameras (time-of-flight, Kinect …)



Laser Scanner

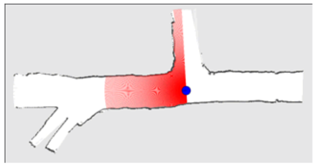
- Measures angles and distances to closest obstacles

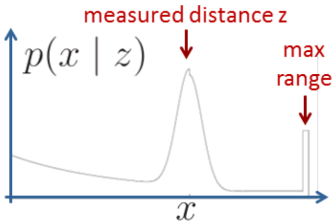
$$\mathbf{z} = (\theta_1, z_1, \ldots, \theta_n, z_n) \in \mathbb{R}^{2n}$$

- Alternative representation: 2D point set (cloud)

$$\mathbf{z} = (x_1, y_1, \ldots, x_n, y_n)^\top \in \mathbb{R}^{2n}$$

- Probabilistic sensor model $p(z \mid x)$





Laser-based Motion Estimation

How can we best align two laser scans?

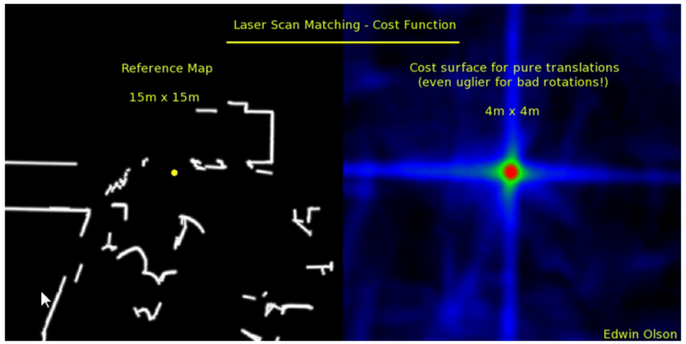
Exhaustive Search

- Convolve first scan with sensor model

- Sweep second scan over likelihood map, compute correlation and select best pose

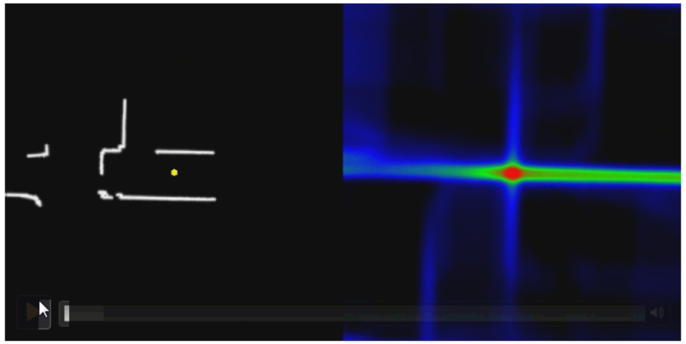
Example: Exhaustive Search [Olson, '09]

- Multi-resolution correlative scan matching
- Real-time by using GPU
- Remember: SE(2) has 3 DOFs



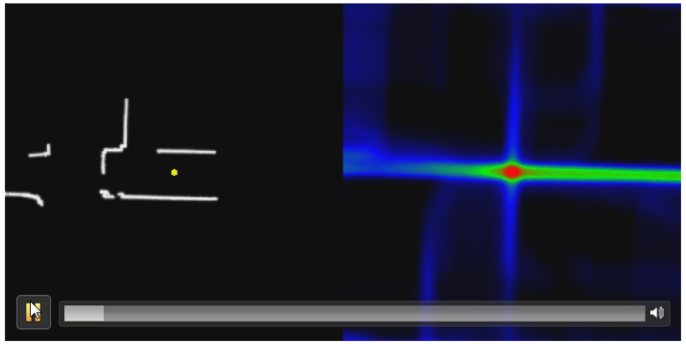
Example: Exhaustive Search [Olson, '09]

- Multi-resolution correlative scan matching
- Real-time by using GPU
- Remember: SE(2) has 3 DOFs

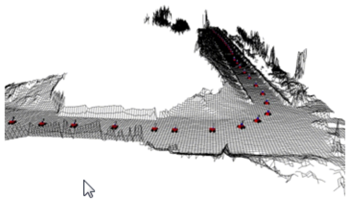


Example: Exhaustive Search [Olson, '09]

- Multi-resolution correlative scan matching
- Real-time by using GPU
- Remember: SE(2) has 3 DOFs



Does Exhaustive Search Generalize To 3D As Well?



Iterative Closest Point (ICP)

- **Given:** Two corresponding point sets (clouds)

$$P = \{\mathbf{p}_1, \ldots, \mathbf{p}_n\}$$
$$Q = \{\mathbf{q}_1, \ldots, \mathbf{q}_n\}$$

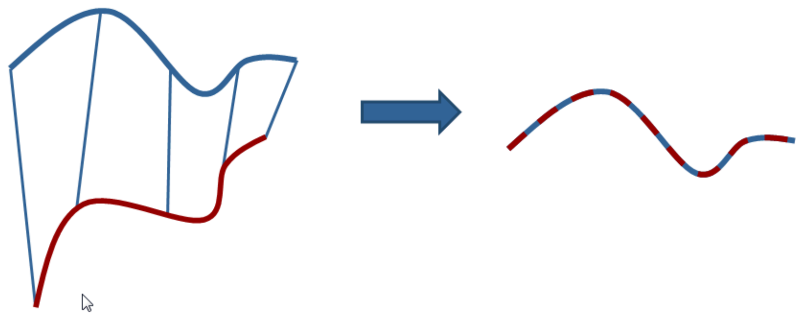
- **Wanted:** Translation $\mathbf{t}$ and rotation R that minimize the sum of the squared error

$$E(R, \mathbf{t}) = \frac{1}{n} \sum_{i=1}^{n} ||\mathbf{p}_i - R\mathbf{q}_i - \mathbf{t}||^2$$

where $\mathbf{p}_i$ and $\mathbf{q}_i$ are corresponding points

Known Correspondences

Note: If the correct correspondences are known, both rotation and translation can be calculated in **closed form**.



Known Correspondences

- **Idea:** The center of mass of both point sets has to match

$$\bar{\mathbf{p}} = \frac{1}{n}\sum_i \mathbf{p}_i \qquad \bar{\mathbf{q}} = \frac{1}{n}\sum_i \mathbf{q}_i$$

- Subtract the corresponding center of mass from every point
- Afterwards, the point sets are zero-centered, i.e., we only need to recover the rotation…

Known Correspondences

- Decompose the matrix

$$W = \sum_i (\mathbf{p}_i - \bar{\mathbf{p}})(\mathbf{q}_i - \bar{\mathbf{q}})^\top = USV^\top$$

  using singular value decomposition (SVD)
- **Theorem**

  If $\operatorname{rank} W = 3$, the optimal solution of $E(R, \mathbf{t})$ is unique and given by

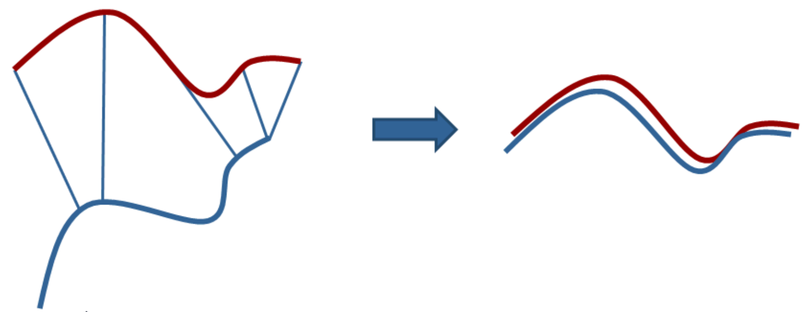
$$R = UV^\top$$

$$\mathbf{t} = \bar{\mathbf{p}} - R\bar{\mathbf{q}}$$

(for proof, see http://hss.ulb.uni-bonn.de/2006/0912/0912.pdf, p.34/35)

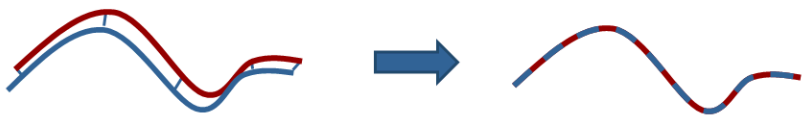
Unknown Correspondences

- If the correct correspondences are not known, it is generally impossible to determine the optimal transformation in one step

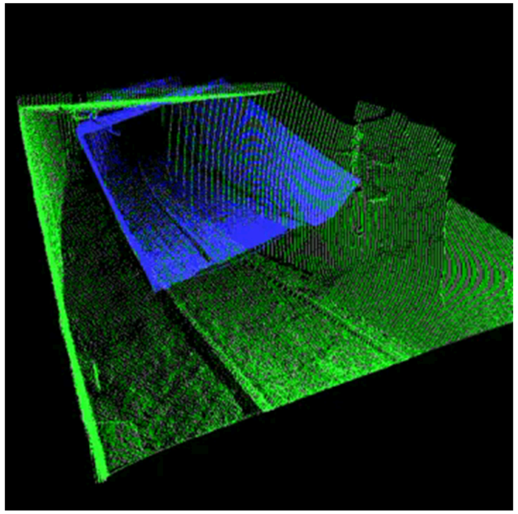


ICP Algorithm
[Besl & McKay, 92]

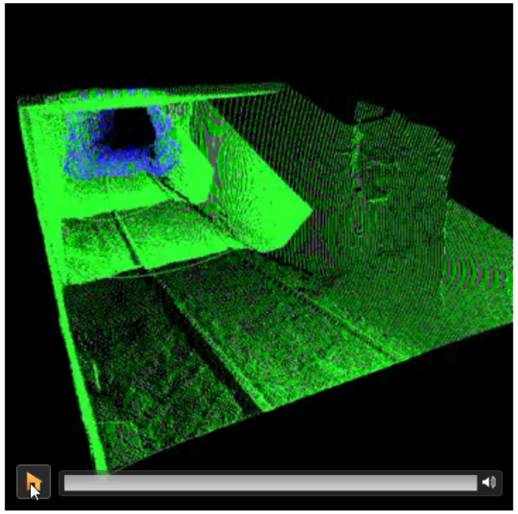
- **Algorithm:** Iterate until convergence
  - Find correspondences
  - Solve for R,t
- Converges if starting position is "close enough"



Example: ICP



Example: ICP



ICP Variants

Many variants on all stages of ICP have been proposed:

- **Selecting** and **weighting** source points
- **Finding** corresponding points
- Rejecting certain (outlier) correspondences
- Choosing an **error metric**
- **Minimization**

Performance Criteria

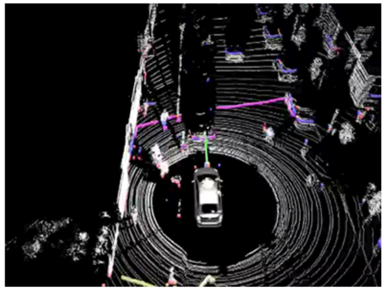
- Various aspects of performance
  - Speed
  - Stability (local minima)
  - Tolerance w.r.t. noise and/or outliers
  - Basin of convergence (maximum initial misalignment)
- Choice depends on data and application

Selecting Source Points

- Use all points
- Uniform sub-sampling
- Random sampling
- Feature-based sampling
- Normal-space sampling
  - Ensure that samples have normals distributed as uniformly as possible

Spatially Uniform Sampling

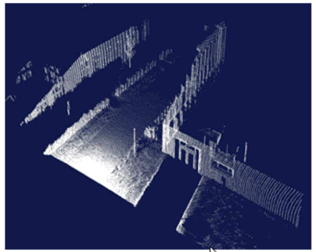
- Density of points usually depends on the distance to the sensor → no uniform distribution
- Can lead to a bias in ICP



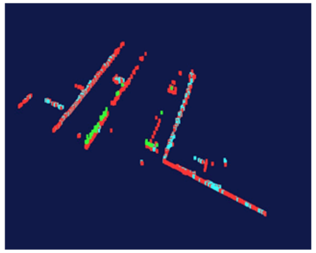
Feature-based Sampling

Detect interest points (same as with images)

- Decrease the number of correspondences
- Increase efficiency and accuracy
- Requires pre-processing

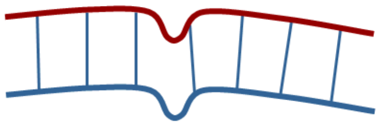


3D Scan (~200.000 Points)

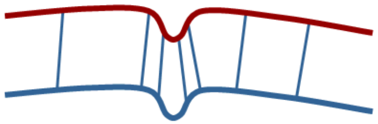


Extracted Features (~5.000 Points)

Normal-Space Sampling



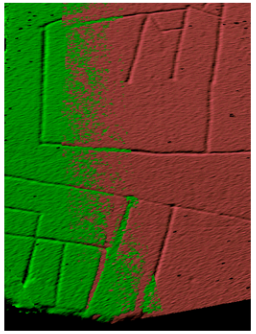
Uniform sampling



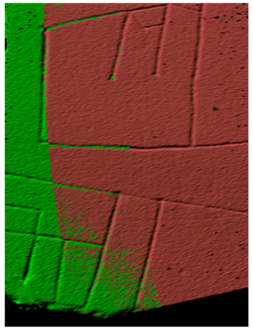
Normal-space sampling

Example: Normal-Space Sampling

Normal-space sampling can help on mostly-smooth areas with sparse features



Random sampling



Normal-space sampling

Selection and Weighting

- Selection is a form of (binary) weighting
- Instead of re-sampling one can also use weighting
- Weighting strategy depends on the data
- Pre-processing / run-time trade-off

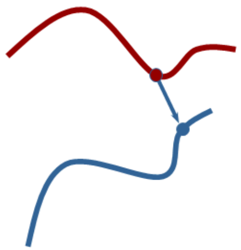
Finding Correspondences

Has greatest effect on convergence and speed

- Closest point
- Normal shooting
- Closest compatible point
- Projection
- Speed-up using kd-trees (or oct-trees)

Closest Point Matching

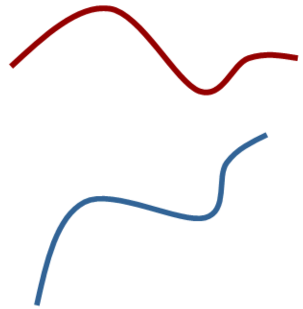
- Find closest point in the other point set
- Distance threshold



- Closest-point matching generally stable, but slow and requires pre-processing

Normal Shooting

- Project along normal, intersect other mesh



- Slightly better than closest point for smooth meshes, worse for noisy or complex meshes

Closest Compatible Point

- Can improve effectiveness of both the previous variants by only matching to **compatible** points
- Compatibility based on normals, colors, …
- In the limit, degenerates to feature matching

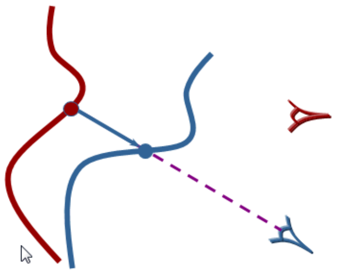
Speeding Up Correspondence Search

Finding closest point is most expensive stage of the ICP algorithm

- Build index for one point set (kd-tree)
- Use simpler algorithm (e.g., projection-based matching)

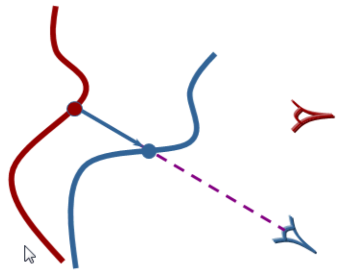
Projection-based Matching

- Slightly worse performance per iteration
- Each iteration is one to two orders of magnitude faster than closest-point
- Requires point-to-plane error metric



Projection-based Matching

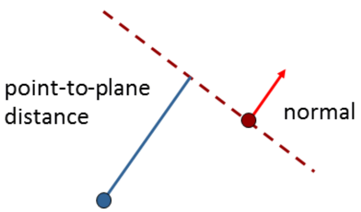
- Slightly worse performance per iteration
- Each iteration is one to two orders of magnitude faster than closest-point
- Requires point-to-plane error metric

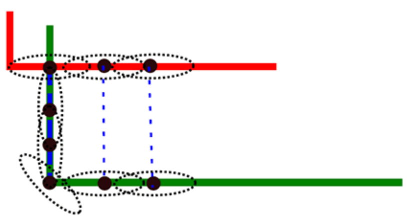


Visual Navigation for Flying Robots 52 Dr. Jürgen Sturm, Computer Vision Group, TUM

Error Metrics

- Point-to-point
- Point-to-plane lets flat regions slide along each other



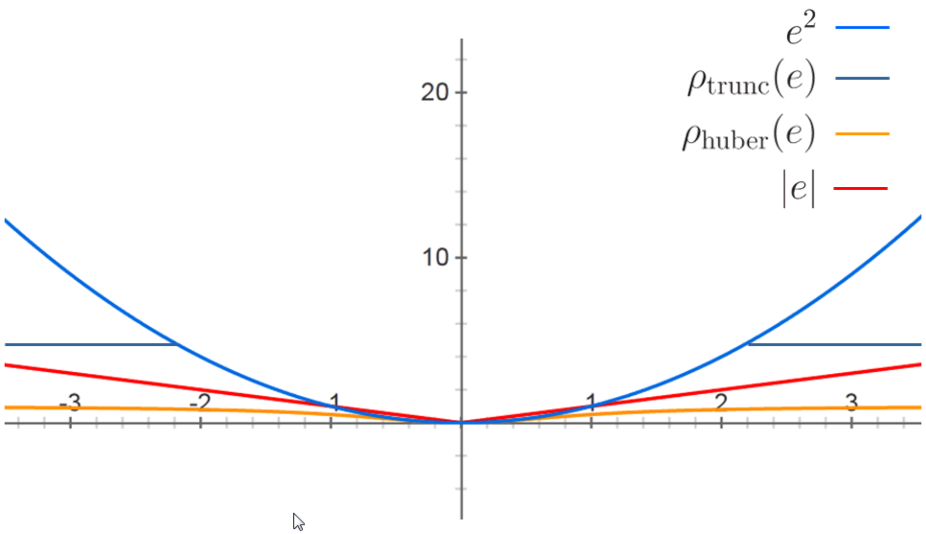


- Generalized ICP: Assign individual covariance to each data point [Segal, 2009]

Minimization

- Only point-to-point metric has closed form solution(s)
- Other error metrics require non-linear minimization methods
  - Which non-linear minimization methods do you remember?
  - Which robust error metrics do you remember?

Robust Error Metrics



Example: Real-Time ICP on Range Images

[Rusinkiewicz and Levoy, 2001]

- Real-time scan alignment
- Range images from structure light system (projector and camera, temporal coding)

ICP: Summary

- ICP is a powerful algorithm for calculating the displacement between point clouds
- The overall speed depends most on the choice of matching algorithm
- ICP is (in general) only locally optimal → can get stuck in local minima

Agenda for Today

- Localization
  - Visual place recognition
  - Scan matching and Iterative Closest Point
- **Mapping with known poses (3D reconstruction)**
  - Occupancy grids
  - Octtrees
  - Signed distance field
  - Meshing

Occupancy Grid

Idea:

- Represent the map $\mathbf{m}$ using a grid
- Each cell is either free or occupied

$$\mathbf{m} = (m_1, \ldots, m_n) \in \{\text{empty}, \text{occ}\}^n$$

- Robot maintains a belief $\text{Bel}(\mathbf{m})$ on map state

Goal: Estimate the belief from sensor observations

$$\text{Bel}(\mathbf{m}) = P(\mathbf{m} \mid \mathbf{z}_1, \ldots, \mathbf{z}_t)$$

Occupancy Grid - Assumptions

- Map is static
- Cells have binary state (empty or occupied)
- All cells are independent of each other

- As a result, each cell m_i can be estimated independently from the sensor observations
- Will also drop index i (for the moment)

Mapping

- **Goal:** Estimate

$$\text{Bel}(m) = P(m \mid z_1, \ldots, z_n)$$

- How can this be computed?

Binary Bayes Filter

- **Prior probability** that cell is occupied $P(m)$ (often 0.5)
- **Inverse sensor model** $P(m \mid z_t)$ is specific to the sensor used for mapping
- The **log-odds representation** can be used to increase speed and numerical stability

$$L(x) := \log \frac{p(x)}{p(\neg x)} = \log \frac{p(x)}{1 - p(x)}$$

Binary Bayes Filter

- **Prior probability** that cell is occupied $P(m)$ (often 0.5)
- **Inverse sensor model** $P(m \mid z_t)$ is specific to the sensor used for mapping
- The **log-odds representation** can be used to increase speed and numerical stability

$$L(x) := \log \frac{p(x)}{p(\neg x)} = \log \frac{p(x)}{1 - p(x)}$$

Binary Bayes Filter using Log-Odds

- In each time step, compute

$$L(m \mid z_{1:t}) = \underbrace{L(m \mid z_{1:t-1})}_{\text{previous belief}} + \underbrace{L(m \mid z_t)}_{\text{inverse sensor model}} + \underbrace{L(m)}_{\text{map prior}}$$

- When needed, compute current belief as

$$\mathrm{Bel}_t(m) = 1 - \frac{1}{1 + \exp L(m \mid z_{1:t})}$$

Clamping Update Policy

- Often, the world is not “fully” static
- Consider an appearing/disappearing obstacle
- To change the state of a cell, the filter needs as many positive (negative) observations
- **Idea:** Clamp the beliefs to min/max values

$$L'(m \mid z_{1:t}) = \max(\min(L(m \mid z_{1:t}), l_{\max}), l_{\min})$$

Sensor Model

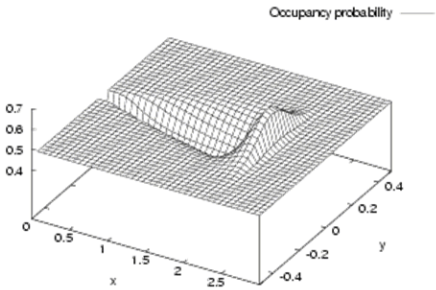
- For the Bayes filter, we need the inverse sensor model

$$p(m \mid z)$$

- Let’s consider an ultrasound sensor
  - Located at (0,0)
  - Measures distance of 2.5m
  - How does the inverse sensor model look like?

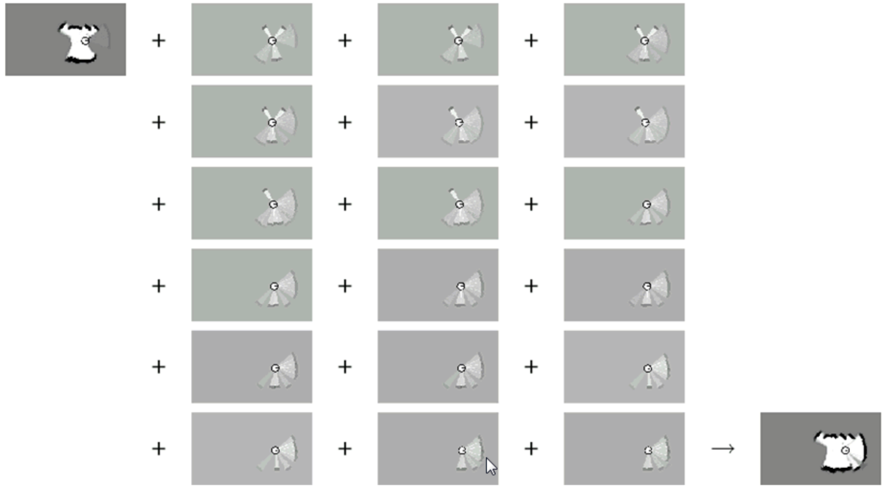
Typical Sensor Model for Ultrasound

- Combination of a linear function (in x-direction) and a Gaussian (in y-direction)



- Question: What about a laser scanner?

Example: Updating the Occupancy Grid



Resulting Map

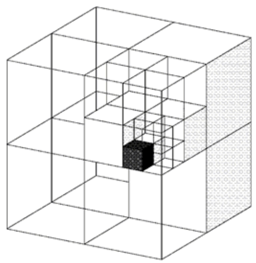
Note: The maximum likelihood map is obtained by clipping the occupancy grid map at a threshold of 0.5

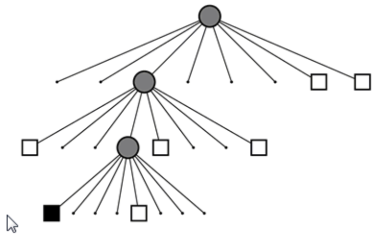
Memory Consumption

- Consider we want to map a building with 40x40m at a resolution of 0.05cm
- How much memory do we need?

Map Representation by Octtrees

- Tree-based data structure
- Recursive subdivision of space into octants
- Volumes can be allocated as needed
- Multi-resolution

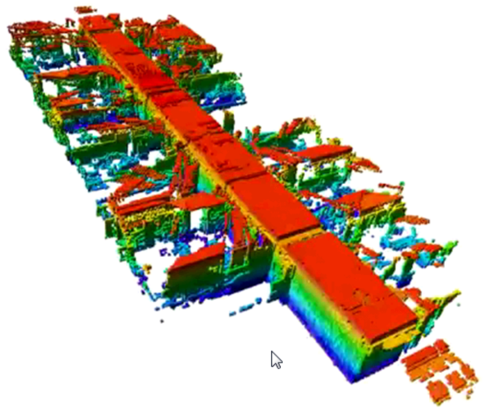




Example: OctoMap

[Wurm et al., 2011]

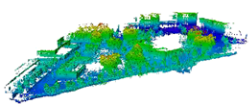
- Freiburg, building 79
  44 x 18 x 3 m^3, 0.05m resolution, 0.7mb on disk



Example: OctoMap

[Wurm et al., 2011]

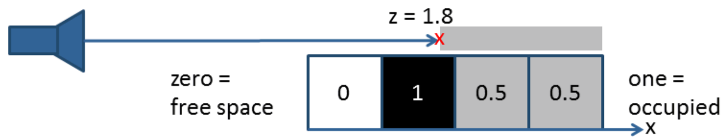
- Freiburg computer science campus
  292 x 167 x 28 m^3, 0.2m resolution, 2mb on disk



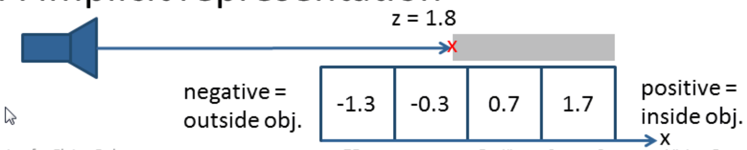
Signed Distance Field (SDF)

[Curless and Levoy, 1996]

- **Idea:** Instead of representing the cell occupancy, represent the distance of each cell to the surface
- Occupancy grid maps: explicit representation



- SDF: implicit representation

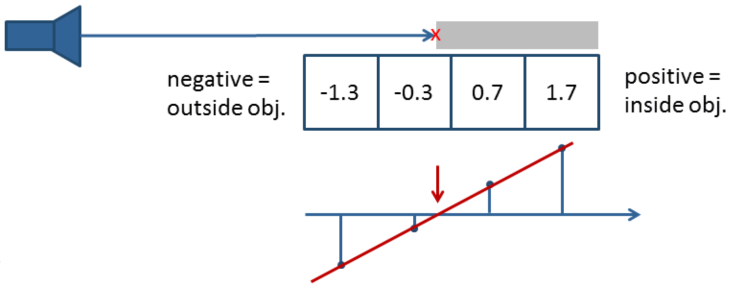


Signed Distance Field (SDF)

[Curless and Levoy, 1996]

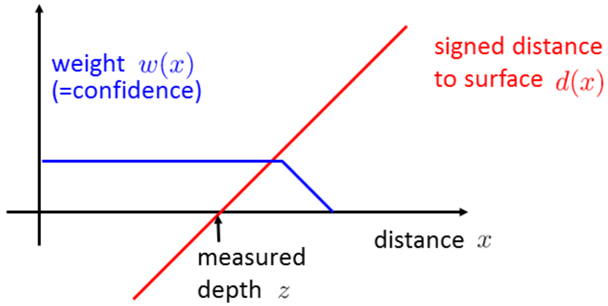
Algorithm:

1. Estimate the signed distance field
2. Extract the surface using interpolation (surface is located at zero-crossing)



Weighting Function

- Weight each observation according to its confidence



- Weight can additionally be influenced by other modalities (reflectance values, …)

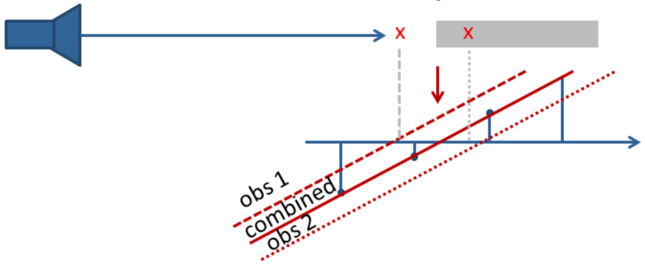
Data Fusion

- Each voxel cell x in the SDF stores two values
  - Weighted sum of signed distances $D_t(\mathbf{x})$
  - Sum of all weights $W_t(\mathbf{x})$
- When new range image arrives, update every voxel cell according to

$$D_{t+1}(\mathbf{x}) = D_t(\mathbf{x}) + w_{t+1}(\mathbf{x})d_{t+1}(\mathbf{x})$$
$$W_{t+1}(\mathbf{x}) = W_t(\mathbf{x}) + w_{t+1}(\mathbf{x})$$

Two Nice Properties

- Noise cancels out over multiple measurements

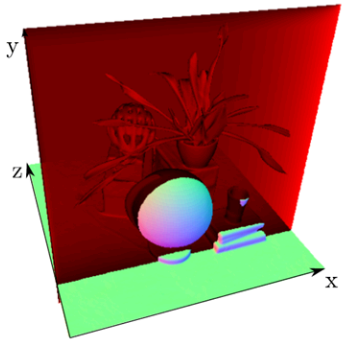


- Zero-crossing can be extracted at sub-voxel accuracy (least squares estimate)

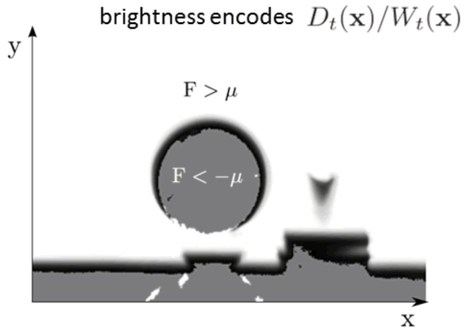
1D Example: $x^* = \dfrac{\sum D_t(x)x}{\sum W_t(x)x}$

SDF Example

A cross section through a 3D signed distance function of a real scene

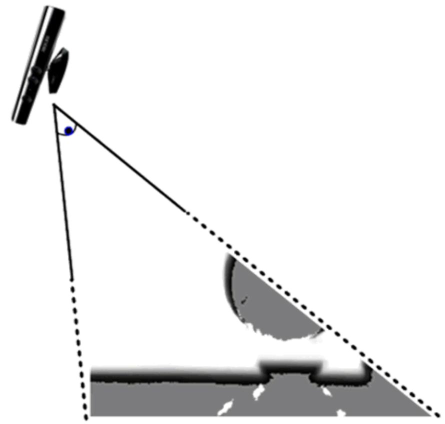


Surface with cross-section

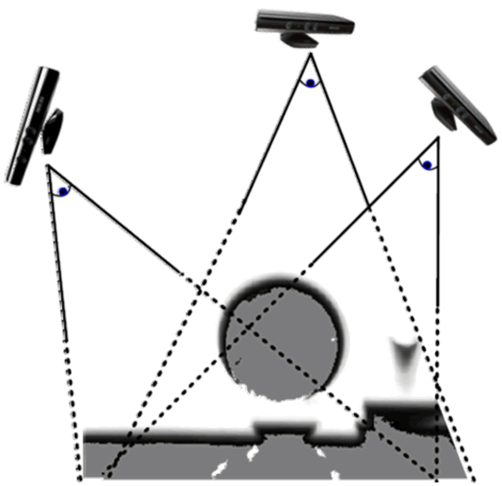


SDF

SDF Fusion



SDF Fusion



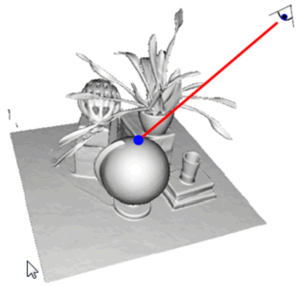
Visualizing Signed Distance Fields

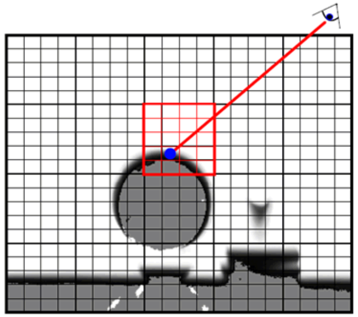
Common approaches to iso surface extraction:

1. Ray casting (GPU, fast)
   For each camera pixel, shoot a ray and search for zero crossing
2. Poligonization (CPU, slow)
   E.g., using the marching cubes algorithm
   Advantage: outputs triangle mesh

Ray Casting

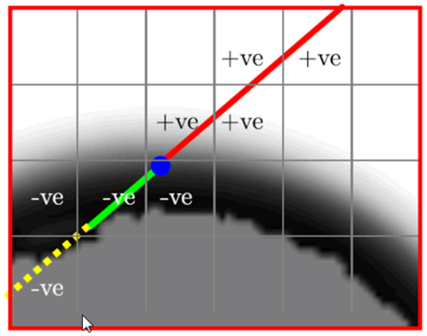
- For each camera pixel, shoot a ray and search for the first zero crossing in the SDF
- Value in the SDF can be used to skip along when far from surface

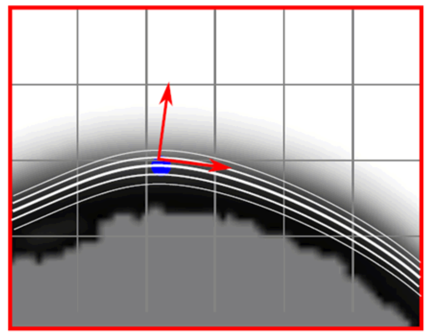




Ray Casting

- Interpolation reduces artifacts
- Close to surface, gradient represents the surface normal

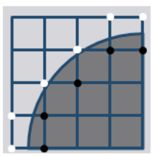


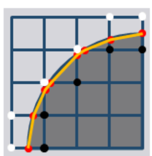


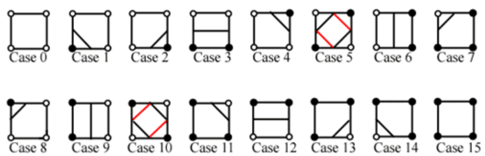
Marching Cubes

First in 2D, **marching squares**:

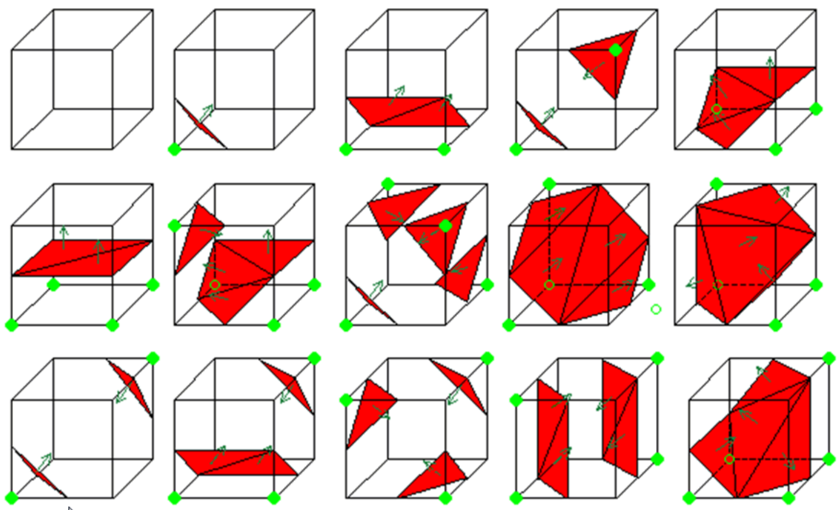
- Evaluate each cell separately
- Check which edges are inside/outside
- Generate triangles according to lookup table
- Locate vertices using least squares







Marching Cubes



KinectFusion

[Newcombe et al., 2011]

- Projective ICP with point-to-plane metric
- Truncated signed distance function (TSDF)
- Ray Casting

KinectFusion

[Newcombe et al., 2011]

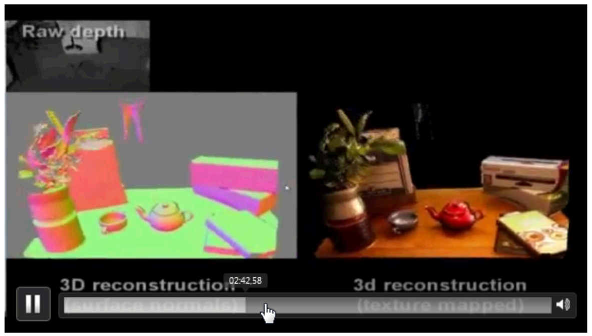
- Projective ICP with point-to-plane metric
- Truncated signed distance function (TSDF)
- Ray Casting

Visual Navigation for Flying Robots 87 Dr. Jürgen Sturm, Computer Vision Group, TUM

KinectFusion

[Newcombe et al., 2011]

- Projective ICP with point-to-plane metric
- Truncated signed distance function (TSDF)
- Ray Casting

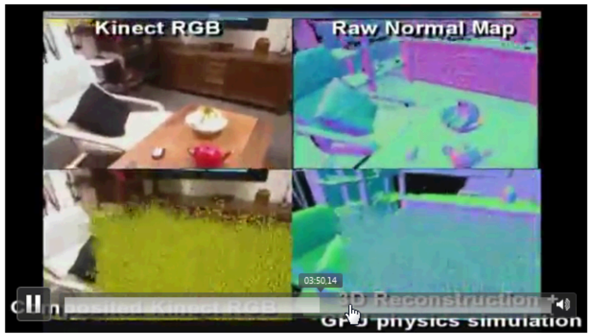


Visual Navigation for Flying Robots 87 Dr. Jürgen Sturm, Computer Vision Group, TUM

KinectFusion

[Newcombe et al., 2011]

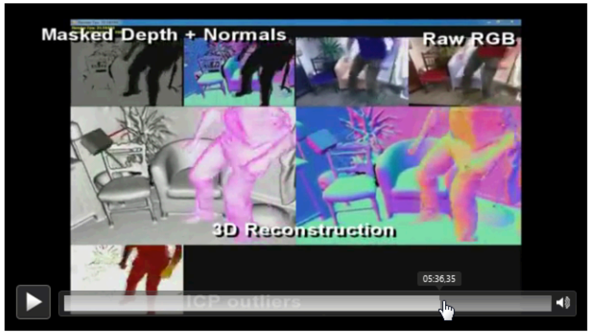
- Projective ICP with point-to-plane metric
- Truncated signed distance function (TSDF)
- Ray Casting



Visual Navigation for Flying Robots 87 Dr. Jürgen Sturm, Computer Vision Group, TUM

KinectFusion
[Newcombe et al., 2011]

- Projective ICP with point-to-plane metric
- Truncated signed distance function (TSDF)
- Ray Casting



Lessons Learned Today

- How to quickly recognize previously seen places
- How to align point clouds
- How to estimate occupancy maps
- How to reconstruct triangle meshes at sub-voxel accuracy