Environment Perception, Mapping, and Motion Planning for Autonomous Mobile Robots

Jonas Ulmen   Roshan Sathyanarayana Shenoy   Naga sai pavan swaroop Ainapurapu   Shrenik Lalani

Institute of Control Systems
Electrical and Computer Engineering
University of Kaiserslautern

9. April 2021

Content

1 Dense Object Descriptors for Object Detection

2 Motion Planning in Autonomous Mobile Robots

3 SLAM

4 References

Computer Vision

Computer vision is the field of computer science that focuses on replicating parts of the complexity of the human vision system and enabling computers to identify and process objects in images and videos in the same way that humans do.

Feature Detection

- Feature detection is the process of computing the abstraction of the image information and making a local decision at every image point to see if there is an image feature of the given type existing in that point.
- An ideal feature detection technique should be robust to image transformations such as rotation, scale, illumination, noise and affine transformations.

Feature Description

- A feature descriptor is an algorithm which takes an image and outputs feature descriptors/feature vectors. Feature descriptors encode interesting information into a series of numbers and act as a sort of numerical "fingerprint" that can be used to differentiate one feature from another.

Machine Learning vs Deep Learning

- In Machine learning, features are extracted manually and passed through a classification algorithm. Involves a lot of work for manual labelling and creation of bounding boxes.

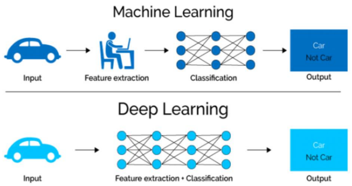


Abbildung: Machine Learning Vs Deep Learning [1]

- Deep learning model involves feeding a computer system lot of data,which it can use to make decision about other data.Generation of descriptors based on neural networks. No need to label and create bounding boxes

Descriptor Learning

Patch Based Descriptor: produce a feature descriptor for each patch defined by a keypoint detector, which can be viewed as direct counterparts for hand-crafted feature descriptors.[2]
Dense Descriptor: use fully-convolutional neural networks to extract dense feature descriptors for the whole image in one forward pass.[3]

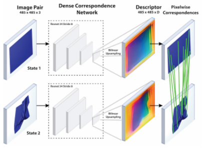


Abbildung: Dense Correspondence generation pipeline [4]

Problem Statement

- Detection of 1D objects like cables for navigation of a mobile robot over an unknown terrain.
- Detection of edges, corner and topography of the object

Dense Object Descriptors

Methodology

- Descriptor matching performed using RGB-D images.

$$f(I) : \mathbb{R}^{H\times W\times 3} \rightarrow \mathbb{R}^{H\times W\times D} \tag{1}$$

- Loss functions: Two images with pixel locations u_a and u_b for image I_a and I_b

$$\mathcal{L}_{\text{matches}}\left(I_a, I_b\right) = \frac{1}{N_{\text{matches}}} \sum_{N_{\text{mathes}}} \left\| f\left(I_a\right)\left(u_a\right) - f\left(I_b\right)\left(u_b\right)\right\|_2^2 \tag{2}$$

$$\mathcal{L}_{\text{non-matches}}\left(I_a, I_b\right) = \frac{1}{N_{\text{non-matches}}} \sum_{N_{\text{non-matches}}} \max\{0, M - \left\| f\left(I_a\right)\left(u_a\right) - f\left(I_b\right)\left(u_b\right)\right\|_2^2\} \tag{3}$$

The final loss for (I_a, I_b) is simply the sum of the two above:

$$\mathcal{L}\left(I_a, I_b\right) = \mathcal{L}_{\text{matches}}\left(I_a, I_b\right) + \mathcal{L}_{\text{non-matches}}\left(I_a, I_b\right)$$

Learned Descriptor Correspondences

- descriptors can be used to grasp the same part of an object (e.g., a shoe) even if the sobject is seen at different camera angles or from different positions.
- works also for different objects, since those still have the same general structure of a "class" and thus descriptors can be consistent even among different class attributes.

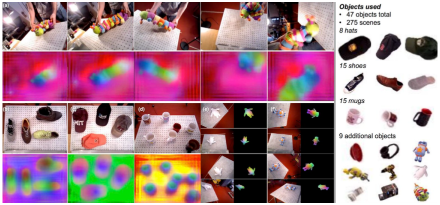


Abbildung: Learned dense Correspondences in different configurations [4]

Other Applications

- capture geometric correspondence across different fabric configura-tions from synthetic RGB images and use them for 2D fabric manipulation.[5]
- can be used to design geometrically structured manipulation policies for grasping.[6]

Offline and Online Path Planning

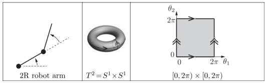
Offline Path Planning

- In offline mode, the path planning task starts from a point where the robot has a complete knowledge about: it's environment with obstacles (given map or global perception through camera for example), its initial position, and its final goal within this environment.
- Generally used for static environment and only when global map of the environment is initially available (given or built).

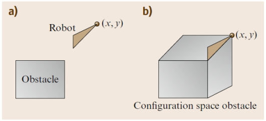
Online Path Planning

- In online mode, path planning is carried out in parallel while moving towards the goal, and perceiving (locally or globally) the environment including its changes with the help of various sensors.
- For dynamic environments, online path planning must be used since the path must be updated according to environment changes.

Configuration Space

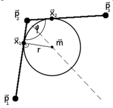


- The configuration of a robot is a complete specification of the position of every point of the robot. The minimum number n of real-valued coordinates needed to represent the configuration is the number of degrees of freedom (dof) of the robot.
- The n-dimensional space containing all possible configurations of the robot is called the configuration space (C-space).



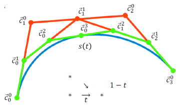
- A robot with complex geometry shape is mapped to a single point in the C-space. Complex geometry shape of robot is converted to a single point by sliding the robot around obstacles.

Linear - Circular Trajectories and Bezier Curves



Linear and Circular Trajectories

- With the help of standard drive kinematics, straight line movements and driving along a circular path with radius bigger than a minimal radius is possible. Also, based on vehicle parameters the radius (r) can be calculated



De Casteljau's Algorithm for Bazier Curve

- Initial points set to control points: $\vec{c_i^0} = P_i$
  Evaluation using recurrent relation: $c_i^{\vec{k+1}} = (1-t)c_i^{\vec{k}} + tc_{i+1}^{\vec{k}}$

Deterministic Algorithms

- A common approach for motion planning consists of representing the configuration space of the robot as a weighted directed graph G=(V,E), where V is a set of possible robot states (locations), and E is a set of edges which represent transitions between these states.

A^*

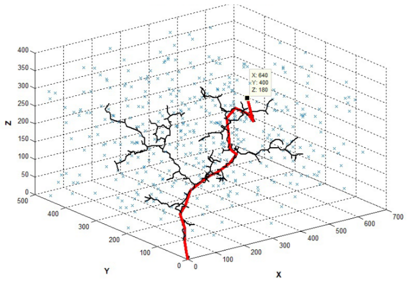
- A^* algorithm picks at each step the node according to a value 'f' which is equal to the sum of 'g' and 'h'. At each step it picks the node having the lowest 'f', and process it.
  $f(n) = g(n) + h(n)$

Visibility Graph

- One approach to complete motion planning is based on representing the complex high-dimensional space C_{free} by one dimensional roadmap R.
- By considering the nodes in an environment as vertices of a graph, the set of all feasible (collision-free) connections between two vertices is called the "Visibility Graph".

Sampling Based Algorithm

- Rapidly-exploring Random Tree (RRT) is an efficient data structure and sampling scheme to quickly search high-dimensional spaces that have algebric constraints (arising from obstacles) and differential constraints (arising from dynamics).
- The key idea of RRT is to bias the exploration toward unexplored portions of the space by sampling points in the state space, and incrementally pulling the search tree towards them.



Other Algorithms

Potential Field Methods

- C-space can also be discretized in a dense and regular grid of configurations. As this grid is generally very wide, this approach requires very powerful heuristic for the path research and potential field method can be a possible solution.
- In Potential Field Approach a manipulator or robot moves in a field of force in such a way that the position to be reached is an attractive pole for a robot and obstacles are repulsive surfaces for a robot.

Optimized Motion Planning

- Optimized Path Planning Using Genetic Algorithm
- Minimum Execution Time Algorithms
- Minimum Energy Algorithms
- Minimum Jerk Algorithms

Navigation

- Success in navigation requires success in the four navigation building blocks: Perception, Localization, Cognition, Motion control.

Challenges to the localization problem: Sensor noises

- Sensor noise causes a constraint on the accuracy of sensor readings in the same state of the environment and, thus, on the amount of usable useful bits from each sensor read.
- All of the additional sources of noise are image jitter, signal gain, blooming and blurring, potentially reducing the useful quality of a color video image.

Challenges to the localization problem:Sensor Aliasing

- The issue raised by sensor aliasing to navigation is that the amount of information is normally inadequate to identify the location of the robot from a single percept reading, even with noise-free sensors.

Simultaneous localization and mapping

- Estimate the pose of a robot and the map of the environment at the same time

SLAM is hard, because

- a map is needed for localization and
- a good pose estimate is needed for mapping

SLAM is a chicken-or-egg problem:

- a map is needed for localization and
- a pose estimate is needed for mapping

SLAM Applications:

SLAM is central to a range of indoor, outdoor, in-air and underwater applications for both manned and autonomous vehicles.Examples:

- At home: vacuum cleaner, lawn mower
- Air: surveillance with unmanned air vehicles

Categorisation of the Problem of SLAM

Types:

- Volumetric Versus Feature-Based
- Topological Versus Geometric Maps
- Known Versus Unknown Correspondence
- Static Versus Dynamic
- Small Versus Large Uncertainty
- Active Versus Passive
- Single-Robot Versus Multirobot SLAM

The Three Main SLAM Paradigms

- Extended Kalman Filters
- Graph-Based Optimization Techniques
- Particle Filters

The Three Main SLAM Paradigms

Extended Kalman Filters Overview

- State prediction (odometry)
- Measurement prediction
- Observation
- Data Association
- Update
- Integration of new landmarks

The key limiting factor in EKF SLAM is the covariance matrix, which takes space (and update time) quadratic in the size of the map.

Graph-Based Optimization Techniques

- A graph-based SLAM approach constructs a simplified estimation problem by abstracting the raw sensor measurements with a probability distribution over the relative locations of the two poses, conditioned to their mutual measurements.
- Graphical SLAM methods have the advantage that they scale to much higher-dimensional maps than EKF SLAM.

The Three Main SLAM Paradigms

Particle Filters

- The key idea is to represent the next probability density function (PDF) of the state variables by a set of random samples or particles with associated weights, and compute estimates based on these samples and weights.
- PF can estimate the system states sufficiently when the number of particles (estimations of the state vectors which evolve in parallel) is large.

Steps to implement particle filter algorithm

- Draw a distribution of even weighted particles.
- Update robot's state in each particle with control command.
- Get observation data
- Update the robot's state estimate by incorporating robot's observation
- Calculate the importance weight of each particle using the difference between actual observation and predicted observation
- Resample particles proportional to their weights.

N. Kumar, "What is the difference between machine learning and deep learning," Oct 2017.

V. Balntas, E. Riba, D. Ponsa, and K. Mikolajczyk, "Learning local feature descriptors with triplets and shallow convolutional neural networks." in *Bmvc*, vol. 1, no. 2, 2016, p. 3.

C. B. Choy, J. Gwak, S. Savarese, and M. Chandraker, "Universal correspondence network," *CoRR*, vol. abs/1606.03558, 2016.

P. Sundaresan, J. Grannen, B. Thananjeyan, A. Balakrishna, M. Laskey, K. Stone, J. E. Gonzalez, and K. Goldberg, "Learning rope manipulation policies using dense object descriptors trained on synthetic depth data," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 9411–9418.

A. Ganapathi, P. Sundaresan, B. Thananjeyan, A. Balakrishna, D. Seita, J. Grannen, M. Hwang, R. Hoque, J. E. Gonzalez, N. Jamali, *et al.*, "Learning to smooth and fold real fabric using dense object descriptors trained on synthetic color images," *arXiv preprint arXiv:2003.12698*, 2020.

P. Florence, L. Manuelli, and R. Tedrake, “Self-supervised correspondence in visuomotor policy learning,” *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 492–499, 2019.