Classifying Obstacles and Exploiting Knowledge about Classes for Efficient Humanoid Navigation

Peter Regier, Andres Milioto, Philipp Karkowski, Cyrill Stachniss, Maren Bennewitz

Presenter: Jeff (Jiaheng) Hu
Terminologies

• Encoder & decoder

• A* search
Task: navigate in a complex environment

• What is special about humanoid robot navigation?
  Differential drive robot  Humanoid robot

• How do we interact with obstacles?
  • Push? Step over? Pick up?
System Overview
Convolutional Neural Network

- RGB picture $\rightarrow$ semantic picture $\rightarrow$ semantic pointcloud

- Evaluate loss using mean intersection over union (mIoU) + mean average precision (mAP)

$$mIoU = \frac{1}{C} \sum_{i=1}^{C} \frac{tp_i}{tp_i + fp_i + fn_i},$$

$$mAP = \frac{1}{C} \sum_{i=1}^{C} \frac{1}{11} \sum_{r \in \{0, 0.1, \ldots, 1\}} p_i(r),$$
Convolutional Neural Network

- Eight classes:
  - "balls", "books", "boxes", "cars", "dolls", "stuffed toys", "toy blocks", and "background"

- Architecture:
  - Based on ResNet and ENet
  - Non-bottlenecks
    - High speed
    - Relatively low accuracy
  - Different dilation rate
Training (Generating data)

- Crawls pictures from Google
- Remix them and change background to generate new data
- Pre-trained network from Bonnet’s library
- Focus on similar objects
Plan & Execute

- One action per object (depends on the robot)

<table>
<thead>
<tr>
<th>Object_class</th>
<th>Action type</th>
</tr>
</thead>
<tbody>
<tr>
<td>balls</td>
<td>push</td>
</tr>
<tr>
<td>toy blocks</td>
<td>step over</td>
</tr>
<tr>
<td>boxes, books</td>
<td>step onto</td>
</tr>
<tr>
<td>stuffed toys, dolls, cars</td>
<td>pick up</td>
</tr>
</tbody>
</table>

- Certain cost for executing each action
- A* search based on the modified cost
- Footstep planning if necessary
- Update and re-plan
Video Demo

- https://www.youtube.com/watch?v=WO94iXT3V1I
Innovation & Key points

- Exploiting knowledge about obstacle classes
- Lightweight CNNs through non-bottlenecks
- Innovative way of generating real-world training data
- Easy to incorporate with other techniques
  - Foot step planning
  - Whole-body motion planning
  - Multi-contact planning
Potential Problem (Personal Thoughts)

• Movement is not continuous
  • Has to update and re-plan every frame

• Directly placing the object behind the robot may not be a good idea
  • What if the robot is not allowed to relocate the object?

• Need an expert to set object – action correspondence
  • How to generalize?

• Assume the same types objects are of similar size
  • What if there’s a big box that the robot can’t step onto? How about cost?
Potential Improvement  (Personal Thoughts)

• Directly generate semantic point cloud
  • Should improve accuracy and speed simultaneously

• D* (field D*) instead of A* when re-planning
  • Improve speed when the map is large

• Increase the default weight of unexplored territories
Thank you!