A Simple Person-Following Robot

Nansong Yi, Thomas Wang, Ollin Boer Bohan

Introduction

We built a simple person-following robot. The robot is a small rover using the UW MuSHR design (built on the MIT RACECAR base) and is programmed with ROS. The only sensor is an intel RealSense D435 stereo camera. Person tracking uses YOLOv3 for detection and simple heuristics (bounding box overlap and visual similarity) for tracking the human subject across frames. Control is entirely reactive, based on the lateral position of the subject in the camera frame and the estimated depth from the RealSense. The robot is capable of indoor and outdoor following of a single subject indefinitely (under favorable conditions).

Motivation

Person-following is a subproblem of the larger goal of natural human-robot interaction. Any mobile robot that interacts with humans successfully will need to be aware of humans around it, follow their directions, and avoid any embarrassing collisions. Most personal robots could or do benefit from awareness of humans (e.g. the Segway miniPLUS follows you when you're not using it; the Skydio R1 follows you while filming; robot vacuum cleaners benefit by walking into humans).

Additionally, the task is simple enough to be implemented within the allotted timeframe while complex enough to demonstrate nontrivial behavior.

Related Work

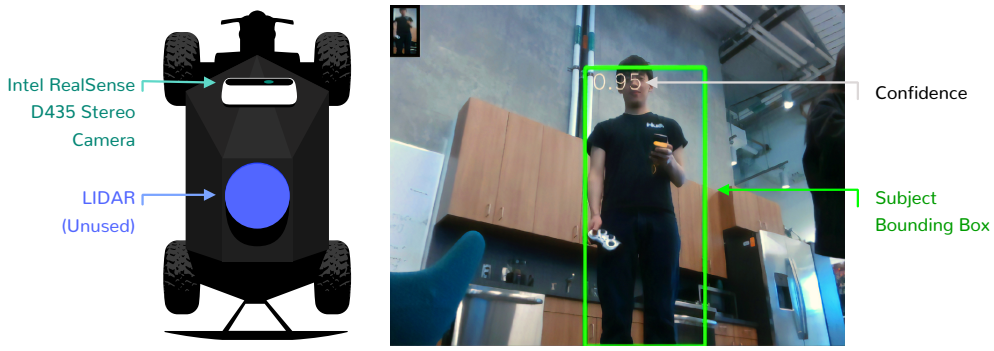
Person-following is a straightforward application of modern tracking algorithms to a mobile robot platform, and, as such, there have been several implementations of this concept on various platforms. For the most part, however, these implementations have had minimal obstacle avoidance capability and have used classical computer vision approaches rather than CNNs for the tracking component.

- Human Following on ROS Framework A Mobile Robot
- Building a human following robot using RealSense camera
- robot-human-follower

The following approach does use CNNs (online, rather than pre-trained) for person tracking, but is closed-source:

- Integrating Stereo Vision with a CNN Tracker for a Person-Following Robot

Methodology



Perception

The foundation of our perception system is the `darknet` ROS package, which provides YOLOv3 tracking boxes at 30hz. From these per-frame tracking boxes we filter out high-confidence person boxes, discard implausibly small or large boxes, and decide on an initial match based on closeness to the camera.

Distance from the person is determined by masking the stereo camera's depth output to the bounding box area and selecting a central value.

Once an initial match is determined, the system compares matches on each subsequent frame to the existing match and ranks the potential matches according to visual similarity (using a structural similarity metric on the cropped region) as well as bounding box overlap.

The tracked bounding box (and its confidence) are sent to an additional node for visualization.

Control

The robot uses a simple reactive control system. Speed is determined based on the estimated depth from the tracked target; the speed is positive if the depth is too great, and negative if the depth is too small. The minimum torque on the robot is too high to follow a walking human precisely (the robot must either be not moving, or moving at a fast walking pace), so there is a "safe zone" of distances in which the robot will not move forward or backward.

Steering angle is determined proportional to the subject's lateral position in the image; the angle is selected such that the robot will end up facing the subject after driving forward, and is continually adjusted during movement. Angles are reversed when driving backwards (to keep the subject in the frame).

Since the perception input is noisy (and detection quality decreases when the robot is moving, due to motion blur), control signals are smoothed slightly to avoid stopping and starting each time the perception drops out.

Evaluation

We conducted indoor and outdoor testing of the final system and observed the following (qualitative) results:

- Under ideal circumstances, the robot is capable of following indefinitely (until the battery runs out).
- Indoors, the most common problem is getting stuck on obstacles (for example, chair legs). The robot's camera is oriented upwards to make detection easier, so proximate obstacles are difficult to handle visually. In simple spaces this is not a problem, since the path traversed by humans is typically obstacle free.
- Outdoors, the most common problem was missed detections due to the lack of dynamic range in the camera (which yielded silhouette-like images with low prediction confidence). Tuning detection thresholds to allow less-probable matches improved this behavior, but potentially at the risk of following false-positive detections (e.g. reflections or human-like objects).
- One interesting result: the robot detects human robots (e.g. the PR2) as people, with moderate confidence.

Conclusion and Future Work

We have demonstrated a simple person-following robot capable of following a subject under typical circumstances. Potential improvements to robustness include:

- **Integration of wheel odometry**: odometry can be used to enhance our system in two key ways; first, it can be combined with a history of visual tracking boxes to perform motion prediction for the tracked subject, which will improve tracking in cases of occlusion or poor visibility. Second, odometry can be used for simple obstacle response–if odometry reports that the robot is stuck, the robot can execute a simple fixed action pattern for recovery (say, driving backwards a bit, then laterally).
- **Visual obstacle avoidance:** the stereo depth map can be used for simple reactive obstacle avoidance (by trying to select a direction such that the central lane has high depth values), or as part of a more sophisticated SLAM system (e.g. ORB-SLAM2) for advanced obstacle avoidance and multi-frame planning.

Team Member Contributions

We developed everything as a group. Although all team members have modified all parts of the code, particular focuses were:

- **Nansong** worked on visualization and the initial tracking code
- **Thomas** worked on visual similarity and improving controller robustness
- **Ollin** wrote the initial controller and also set up initial drafts for the poster and report.