Hector SLAM for robust mapping in USAR environments

ROS RoboCup Rescue Summer School Graz 2012
Stefan Kohlbrecher (with Johannes Meyer, Karen Petersen, Thorsten Graber)
Outline

- Introduction
  - Team Hector
- Requirements
- Hector Mapping
  - Overview
  - Attitude Estimation
  - 2D SLAM
- Hector SLAM Tools
  - GeoTiff
  - Trajectory Server
  - Map Server
- Hector Elevation Mapping
- Examples
- Conclusion
• Team Hector is part of the RTG1362: “Cooperative, Adaptive and Responsive Monitoring in Mixed Mode Environments”
Example
Monitoring in Normal Operation
Example
Some Monitoring Elements and Channels Knocked-Out
Motivation
Deployment of Additional Equipment (Robots, Sensors)

→ Motivates fundamental research questions being addressed by our RTG
Team Hector

- Hector: Heterogeneous Cooperating Team of Robots

- Established in Fall 2008

- Transition from RoboFrame to ROS as the central middleware since late 2010

- You can also like/follow us on facebook: http://www.facebook.com/TeamHectorDarmstadt ;)

Hector SLAM Tutorial | Stefan Kohlbrecher | ROS Workshop Graz | 22/08/2012
Team Hector


3rd place (out of 12 & 16 teams) at the SICK Company’s Robot Day, October 2009 & 2010, Waldkirch

2nd place (out of 27) “Best in Class Autonomy” at RoboCup 2010, Singapore

Winner and “Best in Class Autonomy” at Robocup 2011 GermanOpen

2nd place “Best in Class Autonomy” at RoboCup 2011, Istanbul
Requirements – SLAM in USAR environments

- **SLAM**
  - Map the environment
  - Localize Robot
  - Realtime capable

- **Harsh Terrain**
  - Full 6DOF pose estimation
  - Cannot rely on (wheel/drivetrain) odometry
  - Robustness

- **Primary Mission is search for victims/exploration**
  - Mapping/Localization should not interfere

- Georeferenced Map with robot trajectory and objects of interest
  - Saving GeoTiff maps
Gmapping (2D SLAM, Rao-Blackwellized particle filter)
- Odometry needed (can be simulated through separate scanmatcher)
- Robot pose and map data “jump” relative to map frame
- Does not leverage high scan rates of modern LIDARs

6D SLAM
- Requires acquisition of 3D Laser Scans of the environment
  - Impairs mobility of platform
- Sensitive to correct parameter settings

Visual SLAM
- Not robust enough
- High CPU load
- Promising improvements lately
Hector Mapping - System Overview

- High-level Control
- Localization and Mapping
- Perception
- Navigation
- Drivers

Hector SLAM Tutorial | Stefan Kohlbrecher | ROS Workshop Graz | 22/08/2012
Hector Mapping - System Overview

- **Drivers**: Motors/Sensors, Controller
- **Exploration**
- **Navigation**: Path Planning
- **High-level Control**: Costmap
- **Localization and Mapping**: Elevation Mapping, 2D Mapping
- **Perception**: Object Detection, Object Tracking, Pose Estimation, Laser, Kinect, Cameras
- **Task Execution**: Mission Modeling
- **Controller**: HRI

Hector SLAM Tutorial | Stefan Kohlbrecher | ROS Workshop Graz | 22/08/2012
Hector Mapping - System Overview

This Talk

1. Costmap
2. Task Execution
3. Mission Modeling
4. HRI
5. Exploration
6. Path Planning
7. 2D Mapping
8. Elevation Mapping
9. Pose Estimation
10. Perception
   - Localization and Mapping
   - Object Detection
   - Object Tracking
   - Laser
   - Kinect
   - Cameras
11. Navigation
   - Motors / Sensors
12. Controllers
13. High-level Control

Hector SLAM Tutorial | Stefan Kohlbrecher | ROS Workshop Graz | 22/08/2012
Final Mission RoboCup 2012

- 4 Victims found (3 autonomous, 1 teleop at the end)
- 35 QR codes detected and mapped
- >95% of Arena mapped
- Not a single broken map during missions at RoboCup 2012
Hector Mapping – Recent Results
Hector Mapping – Recent Results

- RGB-D Camera
  - 4m range
  - 30Hz

- Thermal Camera
  - 10Hz
  - 160x120 resolution

- Hokuyo UTM-30LX LIDAR
  - 30m range
  - 40Hz scan rate
  - roll/pitch-stabilized

- Hokuyo URG-04LX LIDAR
  - 4m range
  - 10Hz scan rate

- CO2 Sensor

- Navigation & Control Unit
  - 3 Axis Accelerometer
  - 3 Axis Gyroscope
  - 2 Axis Magnetometer
  - Barometer

- Bi-Directional Audio

- Wheel Encoders (one per wheel)

Processing:
- Geode 500 MHz (low level control)
- Core 2 Duo 2.4 GHz
- Nvidia CUDA GPU optional
Main SLAM node: `hector_mapping`

Main Inputs:
- Scan data on the “/scan” topic
- Transformation data via `tf` (see next slides)

Main Outputs:
- Map on the “/map” topic
- `tf “map” -> “odom” transform` (yielding pose of the robot in the map)
Hector Mapping – Coordinate frames

- `/odom` frame not needed, mainly for compatibility with gmapping/ROS
- `/base_stabilized` frame needed for transformation of LIDAR data
- Height estimation non-trivial
- See also Setup Tutorial
Roll/Pitch Estimation of platform/LIDAR required

- Use IMU for attitude estimation
- We provide the `hector_imu_attitude_to_tf` node
  - Provides `base_stabilized -> base_link` transform
- LIDAR not rigidly mounted to `base_stabilized` frame should be transparent provided correct robot/tf setup
- Best results if LIDAR is actuated and kept approximately level
Map is represented by a 2D grid holding the (log odds representation of the) probability $P_{xy}$ of cell occupancy.

- Access map data on non-integer coordinates using bilinear filtering
  - Only approximative
  - But fast (Trilinear would be other option)
- Cache recently accessed grid points
Hector Mapping – One Iteration

- Receive scan from LIDAR
- Transform scan endpoints into “/base_stabilized” frame
- Throw out endpoints outside cut-offs:
  - laser_z_min_value
  - laser_z_max_value
  - laser_min_dist
  - laser_max_dist
- Perform 2D pose estimation (next slides)
- Update map if robot is estimated to have travelled more than thresholds indicated by
  - map_update_distance_thresh
  - map_update_angle_thresh
Hector SLAM – 2D Pose Estimation (1)

- Set pose estimate, either:
  - Pose from preceding iteration
  - Pose using a tf start estimate
- Iterate:
  1. Project endpoints onto map based on current pose estimate
  2. Estimate map occupancy probability gradients at scan endpoints
  3. Perform Gauss-Newton iteration to refine pose estimate

\[
H = \left[ \nabla M(S_i(\xi)) \frac{\partial S_i(\xi)}{\partial \xi} \right]^T \left[ \nabla M(S_i(\xi)) \frac{\partial S_i(\xi)}{\partial \xi} \right]
\]

Reference:
Gradient-based Optimization can get stuck in local minima
Solution: Use multi-level map representation
Every level updated separately at map update step using scan data
  - No costly downsampling operations between maps anywhere
hector_trajectory_server

- Logs trajectory data based on tf
- Makes Data available as nav_msgs::path via
  - Regularly published topic
  - Service
- Currently used for
  - Visualization of (travelled) robot path
  - GeoTiff node
- Can be used log and visualize arbitrary tf based trajectories
Hector SLAM Tools – Geotiff Node

hector_geotiff

- Provides RC Rescue League compliant GeoTiff maps
- Trigger for saving the map
  - Regular Intervals
  - On Request (topic)
- Runs completely onboard
- Uses ROS services to retrieve
  - Map
  - Travelled path
- Objects of interest can be added using pluginlib based plugins
Hector SLAM Tools – Map Server

map_server

- Retrieves map data via topic
- Makes services for map queries available
- Used on our USAR robot for doing raycasting to project obstacles onto nearest walls
- Already using 3D query so switching to other server type (for example OctoMap based) is straightforward.

```python
# Returns the distance to the next obstacle from the origin of frame point.header.frame_id
# in the direction of the point
#
# All units are meters.

geometry_msgs/PointStamped point
---
float32 distance
```
Hector SLAM – Putting it together

On Hector UGV:

- **hector_pose_estimation**
- **hector_mapping**
- **hector_map_server**
- **hector_trajectory_server**
- **hector_geotiff**

Services:
- /scan
- /poseupdate
- /tf
- /map
- /get_distance_to_obstacle
- /objects
- /map
- /save_geotiff
- /trajectory

Topics:
- **Nodes**

Hector SLAM Tutorial | Stefan Kohlbrecher | ROS Workshop Graz | 22/08/2012
Motivation:

- Elevation map is mandatory for ground robots
- For detection of
  - stairs
  - ramps
  - step fields
  - ...

Proposed Map Representation:

- Two-dimensional array \((x, y)\)
- Height value \((h)\)
- Variance \((\sigma^2)\)
Hector Elevation Mapping

Kalman Filter based Approach:

\[
h(t) = \frac{1}{\sigma_{z(t)}^2 + \sigma_{h(t-1)}^2} \left( \sigma_{z(t)}^2 h(t-1) + \sigma_{h(t-1)}^2 m(t) \right)
\]

\[
\sigma_{h(t)}^2 = \frac{\sigma_{z(t)}^2 \sigma_{h(t-1)}^2}{\sigma_{z(t)}^2 + \sigma_{h(t-1)}^2}
\]

More precisely:

\[
h(t) = \begin{cases} 
  z(t) & \text{if } z(t) > h(t-1) \land dm < c \\
  h(t-1) & \text{if } z(t) < h(t-1) \land dm < c \\
  \frac{1}{\sigma_m^2 + \sigma_{h(t-1)}^2} \left( \sigma_m^2 h(t-1) + \sigma_{h(t-1)}^2 m(t) \right) & \text{else}
\end{cases}
\]

where \( dm \) denotes the Mahalanobis distance:

\[
dm = \sqrt{\frac{(z(t) - h(t-1))^2}{\sigma_{h(t)}^2}}
\]

Reference:
Hector Elevation Mapping

/camera/depth/points

/hector_elevation_mapping

/elevation_map

/poseupdate

Nodes

Topics

Services
Examples - Handheld Mapping System

- Integration of our SLAM system in a small hand-held device
  - Intel Atom processor
  - Same hardware as on our quadrotor UAV
  - Optional connections to GPS receiver, Magnetometer, Barometer for airborne application
Examples - Handheld Mapping System

- RoboCup 2011 Handheld Mapping System dataset
- Small Box with
  - Hokuyo UTM-30LX LIDAR
  - Low Cost (<100$) IMU
  - Atom Z530 1.6 GHz board
- Dataset available at our GoogleCode repository

Embedded Mapping System
RoboCup 2011 Rescue Arena Dataset
11th July 2011 Istanbul
Examples - Handheld Mapping System

- Used to collect data about Wireless Sensor Network (WSN)
- Five minute walk gave 12787 localized WSN signal strength samples
- Much less cumbersome than manual annotation using a floor plan

Reference:
Examples- Unmanned Surface Vehicle

- SLAM System mounted on USV (Unmanned Surface Vehicle)
- Self-Contained, no interconnection with USV (apart from power supply)

USV Experiment
29th August 2010
Claytor Lake, Virginia
Conclusion

- Hector SLAM:
  - Robust SLAM in simulated USAR environments
  - Accurate enough to not need explicit loop closure in many real world environments
  - Lets you focus on your other research topics (path planning, high level planning etc.)
  - Relies on high update rate laser scanners, does not need odometry

- People at Google Munich looking into combining Hector Mapping and Octomap for real 3D support (Repo [here](#))
Appendix: Localization and Mapping
Inertial Navigation System

- Estimation of the **full 3D state** (position, orientation, velocity) of the robot from different sensor sources:
  - Inertial Measurement Unit (IMU)
  - Compass (Magnetic Field)
  - Global Satellite Navigation
  - Altimeter, Range Sensors etc.

### Problems:
- Absolute position is not very accurate or not available at all
- Solution suffers from drift
- Acceleration can lead to significant orientation errors

[Diagram: Strapdown Inertial Navigation Algorithm]
Appendix: Localization and Mapping
Combine SLAM and EKF

- **Our approach**: Couple both localization approaches in a loose manner.

![Diagram of SLAM and EKF integration](image)
Appendix: Localization and Mapping

Navigation Filter

- Sensor information is fused using an **Extended Kalman Filter (EKF)**
- **State Vector**

\[
\tilde{x}_k = (\Omega_k^T, p_k^T, v_k^T, \delta\omega_k^T, \delta a_k^T)^T
\]

- **Orientation**
- **Position**
- **Velocity**
- **Angular rate error**
- **Acceleration error**
- **System Input** (= Inertial Measurements)

\[
u_k = (\omega_k^T, a_k^T)^T
\]

- **Angular rates**
- **Accelerations**
Appendix: Localization and Mapping Integration

Pose Update from SLAM:
- Pose estimates from EKF and SLAM have unknown correlation!
- Solution: Covariance Intersection (CI) approach [5]

\[
(P^+) = (1 - \omega) \cdot P^{-1} + \omega \cdot C^T R^{-1} C
\]

\[
\mu^+ = P^+ \left( (1 - \omega) \cdot P^{-1} \mu + \omega \cdot C^T R^{-1} z \right)^{-1}
\]

with
- Estimated state and covariance (a-priori): \((\mu, P)\)
- Scan Matcher pose and covariance: \((z, R)\)
- Observation matrix \(C\)
- Tuning parameter \(\omega \in [0, 1]\)