Rapyuta: A Cloud Robotics Platform
Gajamohan Mohanarajah, PhD candidate, ETH Zurich
Motivation - I

RoboEarth Database

Action Recipes
Maps
Software Components
Objects

Clients

- Generic Software Components
HAL - Hardware Abstraction Layer
More challenges along the way...

I wish RoboEarth did some some processing

Are you serious?

Can I join too?
Motivation – II

Advances in Mobile Communication Technologies

Onboard Execution

Cloud Execution

Allowed Delay

Data Rate

Data
The Complete Picture!

- RoboEarth Database
- Action
- Recipes
- Maps
- Software Components
- Objects

- RoboEarth Cloud Engine

- Clients
  - HAL

- Generic Software Components
- HAL - Hardware Abstraction Layer
Let's look into details
Details
Other use cases

- Robot Networking
- rosbridge like functionality
- multi master functionality
Turtlebot: Internet Edition

PrimeSense (200$)

ASUS USB-N53 (35$)

ODROID U2 (~90$)

Total price < 550$

Task: Tokyo Big Sight #610, Amazon Ireland server, map it in real time!
Architecture Overview
Visual Odometry

- Process of determining camera pose from image sequence
- Usually feature-based incremental frame to frame matching
- Our approach - keyframe based dense visual odometry
- Dense visual odometry - uses all pixels to estimate the pose (based on Steinbruecker et al. 2011 & Kerl et al. 2013)
- Keyframe - reference frame that is used to estimate relative pose of other frames
Loop Closure and Global Optimization
Global Optimization Algorithm

- Step 1: Build error graph of the problem
  - Find constraints (edges) between all pairs of keyframes (parallelize)
- Step 2: Optimize the global error function
  - Open-source packages - g2o, ceres
Map Merging

- Select a random frame from one map
- Extract features
- Try to match with frames from other maps
Evaluation of Bandwidth

- 3.5 MB/s for every frame (h.264 + FFV1)
- 300-450 KB/s for keyframes only
- Bandwidth is proportional to the speed of the robot
Evaluation

- Parallel optimization speedup more visible on bigger datasets
- Computation time stops decreasing after 8 machines due to communication costs
Demo
Conclusion

- Rapyuta Overview / Architecture
- Cloud based multi-robot 3D mapping system
Thank you for your attention

• Questions?
Why not use a general PaaS for robotics?

- Limitations:
- General: Most supports single process apps
- Google App Engine: Programing APIs, execution time limit
- Heroku: No WebSockets, No inter communication between containers
- OpenShift/Cloud Foundry (New!): General limitations still apply
Other robotic PaaS like frameworks

- Davinci: Single ROS system in the cloud, Hadoop backend for mapping
- UNR-PF (ongoing): Cloud networked robotics platform
Challenges

• Limited computation power on each robot

• Limited bandwidth and unreliable communication

• Much available computation power on the cloud, but not suitable for real-time processing