ANYmal at the ARGOS Challenge

Tools and Experiences from the Autonomous Inspection of Oil & Gas Sites with a Legged Robot

Péter Fankhauser
Remo Diethelm, Samuel Bachmann, Christian Gehring, Martin Wermelinger, Dario Bellicoso, Vassilios Tsounis, Andreas Lauber, Michael Bloesch, Philipp Leemann, Gabriel Hottiger, Dominik Jud, Ralf Kaestner, Linus Isler, Mark Hoepflinger, Roland Siegwart, Marco Hutter
ARGOS Challenge

“Creating the first autonomous robot for gas and oil sites”

http://www.argos-challenge.com
DESCRIPTION OF THE ENVIRONMENT

DESCRIPTION OF THE TESTING GROUND

The performances of the robot systems will be evaluated during the 3rd competition in the same decommissioned unit representative of a typical TOTAL production site, i.e. the UMAD facility.

The facility used for the competition is used as a training site for human operators. It now uses utilities such as water and nitrogen. It is equipped with pumps for circulating these utilities. Leaks of water can be generated. As it can be seen on the picture (Figure 1), it is an outdoor facility.

The important elements from the mobility perspective are detailed below. In particular, the UMAD site is a multi-floor facility without elevator between the floors. The robot system will have to manage this constraint under requirements described in "The Robot System" section (page 18).

A 3D model of the competition site has been provided during the Challenge Kick-Off Meeting, C-KOM (Refer to Appendix D, page 83).
ARGOS Challenge
5 International Teams

AIR-K
Japan

ARGONAUTES
Austria & Germany

FOXIRIS
Spain & Portugal

VIKINGS
France

LIO
Switzerland
ARGOS Challenge

Team LIO
ANYmal – A High-Performance & Versatile Quadrupedal Robot

ANYdrive – A Integrated, Robust, Torque-Controllable Robot Joint

- Fully integrated
- Accurate position & torque control
- Absolute position sensing
- Programmable controller
- Impact robust
- Hollow-shaft
- Water-proof
System Overview

**Locomotion PC**
- State estimation: Legged odometry, ground est.
- Locomotion Control: Trotting, crawling, Free Gait
- ROS interface

**Navigation PC**
- Localization: ICP, Rovio, Tango etc.
- Terrain mapping: Elevation mapping, traversability est.
- Motion planning: Navigation, foothold selection
- Motion planning: Navigation, foothold selection
- Mission execution

**Inspection PC**
- Visual inspection: Gauges, levers
- Thermal inspection
- Audio inspection: Pump & alarm sounds

**Operator PC**
- Remote control UI: Manual, supervised control
- Visualizations: Sensors, robot state, environment
- Mission: Mission creating and protocol

**Hardware Components**
- ANYdrive
- IMU
- Foot contact sensor
- Laser range sensor
- Wide-angle camera
- Zoom camera
- Thermal camera
- Microphone
- Visible & IR light
- Laser range sensor
- Wide-angle camera
- Zoom camera
- Thermal camera
- Microphone
- Visible & IR light

**Networking**
- Ethernet
- WiFi
- Radio
Locomotion

State Estimation

Whole-Body Control

Actuation

Robot

Sensor

Robot state

Joint commands

Joint torques

measurements
Locomotion
State Estimation

Extended Kalman Filter

- No assumption on terrain

Extended Kalman Filter

- No assumption on terrain
- Kinematic measurements (encoders) for legs in contact

Extended Kalman Filter

- No assumption on terrain
- Kinematic measurements (encoders) for legs in contact

Locomotion
State Estimation

Extended Kalman Filter

- No assumption on terrain
- Kinematic measurements (encoders) for legs in contact
- Fused with inertial measurements (IMU)
- Error < 5% over distance

Kinematic measurements

Locomotion
State Estimation

Extended Kalman Filter
- No assumption on terrain
- Kinematic measurements (encoders) for legs in contact
- Fused with inertial measurements (IMU)
- Error < 5% over distance
- Optionally combined with external pose (GPS, laser, vision, etc.)

Locomotion

Whole-Body Control

Locomotion Controller Modules (Loco)

- Virtual model control
- Trajectory optimization
- Zero-Moment Point Traj.

Pose optimization
Whole-Body Control
Contact force distribution

Reflexes
Gait patterns

Robot Controller Manager (Rocoma)

Controller Layer

- Trotting
- Crawling
- Free Gait
- ...

First Emergency Layer

- Emergency Controller A
- Emergency Controller B

Second Emergency Layer

Fail-proof Controller


Locomotion

Free Gait – An Architecture for the Versatile Control of Legged Robots

- Abstraction Layer for Whole-Body Motions (Free Gait API)

Locomotion

Free Gait – An Architecture for the Versatile Control of Legged Robots

- Abstraction Layer for Whole-Body Motions (Free Gait API)
- Robust motion execution in task space

Locomotion
Free Gait – An Architecture for the Versatile Control of Legged Robots

- Abstraction Layer for Whole-Body Motions (Free Gait API)
- Robust motion execution in task space
- Implemented as ROS Action (with frameworks for YAML, Python, C++)

Locomotion

Kindr – Kinematics and Dynamics for Robotics

- C++ library for the consistent handling of 3d position and rotations
- Support for rotation matrices, quaternions, angle-axis, rotation vectors, Euler angles, etc.
- Support for all common operations and includes time-derivates
- ROS interface available
- Based on Eigen, 1000+ unit tests

Navigation

Laser Range Data → Localization → Elevation Mapping

Traversability Est. → Navigation Planning → Execution

Localization

Elevation Mapping

Navigation Planning

Execution
Navigation
Laser-Based Localization (Iterative Closest Point (ICP))

- Point cloud registration for localization in reference map
- Full rotation of LiDAR is aggregated for point cloud
- Use of existing maps or online mapping

Navigation
Elevation Mapping – Dense Terrain Mapping

- Probabilistic fusion of range measurements and pose estimation
- Explicitly handles drift of state estimation (robot-centric)
- Input data from laser, Kinect, stereo cameras, Velodyne etc.

Navigation

Grid Map – Universal Multi-Layer Grid Map Library

Navigation

Grid Map – Universal Multi-Layer Grid Map Library

- 2D circular buffer data structure
  - Efficient map repositioning

Navigation

Grid Map – Universal Multi-Layer Grid Map Library

- 2D circular buffer data structure
  - Efficient map repositioning
- Based on Eigen (C++)
  - Versatile and efficient data manipulation

```c
double rmse =
    sqrt(map["error"].array().pow(2).sum() / nCells);
```
Navigation

Grid Map – Universal Multi-Layer Grid Map Library

- 2D circular buffer data structure
  - Efficient map repositioning
- Based on Eigen (C++)
  - Versatile and efficient data manipulation
- Convenience functions
  - Iterators, math tools, etc.

Navigation

Grid Map – Universal Multi-Layer Grid Map Library

- 2D circular buffer data structure
  - Efficient map repositioning
- Based on Eigen (C++)
  - Versatile and efficient data manipulation
- Convenience functions
  - Iterators, math tools, etc.
- ROS & OpenCV interfaces
  - Conversion from/to images, point clouds, occupancy grids, grid cells

Navigation
Traversability Estimation

Elevation map
Slope
Roughness
Step height

\[ \sum \]

Traversability map


Open Source

github.com/ethz-asl/traversability_estimation
Navigation Planning

- Online navigation planning based on RRT* (OMPL)
- Works with and without initial map
- Continuous for changing environments

Inspection

Visual inspection

- Pressure & Level gauges
- Valves

Thermal Inspection

- Thermal points
- Pumps
- Gas leaks
- Platform alarm

Auditive Inspection

- Zoom-camera
- Microphones (audible and ultra-sonic)
After zooming the manometer to a readable size and before it is possible to read the dial, the manometer has to be de-warping and rotated to a vertical front view. To do this the orientation is determined by matching the image with a known template using SIFT features. If more than four features are matched it is then possible to find the homography matrix which contains information about rotation and perspective. Eventually, with the inverse of the homography the scene image is de-warping and rotated to the desired front view.

Figure 27 Matching and de-warping example. On the left hand side is the known template, the lines indicate matched SIFT features, the green box visualises the homography and the image on the right hand side shows the de-warped and rotated sensor dial, ready to be read.

In case that the SIFT matching does not work, either because the manometer is rotated too much or due to lighting issues, a backup strategy is implemented. The idea is to use the predefined orientation and rotation of the dial to de-warp the dial manually without the SIFT features. However, this method works only if the actual manometer orientation doesn’t deviate too much from the predefined orientation. Otherwise, the result will be a unusable wrong de-warped image.

Figure 28 The left image is the original recorded image and the other one is manually de-warped.

User Interface

Interface for remote control, semi-, and full autonomous operation.

Pán-tilt-zoom, illumination, video recording, and image capturing. This GUI gives full control over the pan-tilt-zoom camera. Additionally, it is possible to enable/disable illumination, to record videos, and to capture images.

We furthermore reconstruct the environment from the multiple cameras around the robot giving the operator an overview of its surrounding (e.g., to search something). The operator can click in the image and select areas of interest where the zoom camera provides a detailed view.

Interface for remote control, semi-, and full autonomous operation.
User Interface

Interface for remote control, semi-, and full autonomous operation.

- Situational camera
- 3D view (RViz)
- Mission control & protocol
- Other modules

Inspection cameras
Robot actuators & sensors
Error protocol

Autonomous Systems Lab
User Interface

This GUI gives full control over the pan-tilt-zoom camera. Additionally, it is possible to enable/disable illumination, to record videos and to capture images.

We furthermore reconstruct the environment from the multiple cameras around the robot giving the operator an overview of its surrounding (e.g. to search something). The operator can click in the image and select areas of interest where the zoom camera provides a detailed view.
User Interface

Bandwidth Considerations

- Only critical data is transmitted by default (robot state and position)
- Other data is transmitted on demand (video, maps, etc.)
- Separation of onboard TF and operator TF
- Connection status node monitors WiFi status and triggers recovery behavior
User Interface
ANYping Indicator

- Indicates PC network availability in Ubuntu menu bar
User Interface

Pose Graph

- Pose graph for inspection, special maneuvers (e.g. stairs), docking station etc.
- Visualization and interactive editing of pose graph
- Continuous updating and (re-)planning on pose graph during mission
User Interface
Mission Creation

- Task-level state machine (C++ library, similar to SMACH)
- State machine defined in YAML format
- Common building blocks to facilitate construction
User Interface
Mission Creation

- Task-level state machine (C++ library, similar to SMACH)
- State machine defined in YAML format
- Common building blocks to facilitate construction
- Typical missions programmed in 5–20 minutes
RQT Multiplot Plugin & Variant Topic Tools

- C++ library (alternative to rqt_plot)
- Multiple plots in one window
- Edit, save, and load configurations
- Live plotting or load rosbags
RQT Multiplot Plugin & Variant Topic Tools

- C++ library (alternative to rqt_plot)
- Multiple plots in one window
- Edit, save, and load configurations
- Live plotting or load rosbags
- Easy to setup configurations
RQT Multiplot Plugin & Variant Topic Tools

- C++ library (alternative to rqt_plot)
- Multiple plots in one window
- Edit, save, and load configurations
- Live plotting or load rosbags
- Easy to setup configurations
- x- and y-axis freely configurable

Open Source

github.com/ethz-asl/rqt_multiplot_plugin

github.com/ethz-asl/variant
Software Tools – How We (Try) To Keep Things Smooth

- All developers and robots same setup
  - Ubuntu 14.04 LTS, ROS Indigo
Software Tools – How We (Try) To Keep Things Smooth

- All developers and robots same setup
  - Ubuntu 14.04 LTS, ROS Indigo
- Software version control with Git
  - Bitbucket & GitHub
Software Tools – How We (Try) To Keep Things Smooth

- All developers and robots same setup
  - Ubuntu 14.04 LTS, ROS Indigo

- Software version control with Git
  - Bitbucket & GitHub

- Conventions for package structure, format, naming, and code style
  - github.com/ethz-asl/ros_best_practices/wiki
Software Tools – How We (Try) To Keep Things Smooth

- All developers and robots same setup
  - Ubuntu 14.04 LTS, ROS Indigo
- Software version control with Git
  - Bitbucket & GitHub
- Conventions for package structure, format, naming, and code style
  - [github.com/ethz-asl/ros_best_practices/wiki](https://github.com/ethz-asl/ros_best_practices/wiki)
- Extensive use of simulation
  - Gazebo
Software Tools – How We (Try) To Keep Things Smooth

- All developers and robots same setup
  - Ubuntu 14.04 LTS, ROS Indigo

- Software version control with Git
  - Bitbucket & GitHub

- Conventions for package structure, format, naming, and code style
  - [github.com/ethz-asl/ros_best_practices/wiki](https://github.com/ethz-asl/ros_best_practices/wiki)

- Extensive use of simulation
  - Gazebo

- Visualizing as much as possible
Software Tools – How We (Try) To Keep Things Smooth

- Lots of tests on hardware
  - Weekly “shakeouts” for defined tasks
  - Lots of demos
Software Tools – How We (Try) To Keep Things Smooth

- Lots of tests on hardware
  - Weekly “shakeouts” for defined tasks
  - Lots of demos

- Continuous Integration
  - Jenkins
  - Unit tests (after each change)
  - ROS integration tests (at night)
Software Tools – How We (Try) To Keep Things Smooth

- Lots of tests on hardware
  - Weekly “shakeouts” for defined tasks
  - Lots of demos

- Continuous Integration
  - Jenkins
  - Unit tests (after each change)
  - ROS integration tests (at night)
Software Tools – How We (Try) To Keep Things Smooth

- Lots of tests on hardware
  - Weekly “shakeouts” for defined tasks
  - Lots of demos

- Continuous Integration
  - Jenkins
  - Unit tests (after each change)
  - ROS integration tests (at night)

- Logging (rosbag)
  - All important information is always logged
  - Review logs with RViz and RQT Multiplot
Conclusion

- Introduced 10 open-source packages, 250+ internal packages
- Coordination of a big team is hard
- Good naming is important
- ROS as “glue”
- WiFi is often problematic
- Reliability is crucial
Thank You

Open-Source Software

github.com/ethz-asl

github.com/leggedrobotics

Péter Fankhauser
pfankhauser@ethz.ch

Samuel Bachmann  Dominic Jud
Dario Bellicoso  Ralf Kaestner
Michael Bloesch  Bruno Kaufmann
Remo Diethelm  Philipp Krüsi
Christian Gehring  Andreas Lauber
Mark Hoepflinger  Philipp Leemann
Gabriel Hottiger  Konrad Meyer
Marco Hutter  Roland Siegwart
Jemin Hwangbo  Vassilios Tsounis
Linus Isler  Martin Wermelinger

www.rsl.ethz.ch
www.asl.ethz.ch