Tech United Eindhoven @Home Team Description 2014


Eindhoven University of Technology,
Den Dolech 2, P.O. Box 513, 5600 MB Eindhoven, The Netherlands
http://www.techunited.nl, techunited@tue.nl

Abstract. This paper provides an overview of the hardware and software of the Tech United Eindhoven RoboCup@Home team. In hardware, our main developments concern the development of a new platform. Our main software improvements are in 3D navigation, whole-body motion planning, perception and world modeling.

1 Introduction

Tech United Eindhoven is the RoboCup team of the Eindhoven University of Technology, competing in the Middle Size League and the @Home League. Tech United has been competing in the @Home league for three years, scoring a seventh place at the 2013 RoboCup German Open and a third place at RoboCup 2013 in Eindhoven. Tech United Eindhoven consists of PhD and MSc students and staff members from different departments within the Eindhoven University of Technology.

This Team Description Paper is part of the qualification package for RoboCup 2014 in João Pessoa and describes the current status of the @Home activities of Tech United Eindhoven. First, the hardware and software of the AMIGO robot platform and its successor, SERGIO, will be introduced. This will be followed by a description of this year’s main research improvements in the fields of navigation, whole-body planning and control, perception and world modeling.

2 Tech United Eindhoven @Home Robots

During the past three years the AMIGO robot, which will be briefly discussed in Section 2.1, has competed in the @Home league. Based on our experiences with this robot, a new platform has been developed, which will be introduced in Section 2.2. All CAD drawings and electrical schemes of the Tech United robots can be found on Robotic Open Platform\(^1\). Some general remarks about our software design will be made in Section 2.3.

2.1 The AMIGO Hardware

AMIGO is an acronym for: Autonomous Mate for IntelliGent Operations (see Figure 1). The base platform of AMIGO has originally been designed for a middle-size league soccer robot. The main advantage an omniwheel platform has is that it can move instantaneously in any direction. Four omniwheels are used to maximize stability and the available space for peripheral equipment.

On top of the base platform, the upper body of the AMIGO is mounted using a ball screw spindle mechanism to adjust the height of the upper body. For manipulation, the upper body is equipped with two 7-Degree-of-Freedom (DoF) Philips robotic arms.

For navigation and object recognition, the AMIGO relies on various sensors. On the base of the robot, a Hokuyo laser range finder (LRF) is used for localization and obstacle avoidance. A second LRF, which is mainly used for people detection, is placed on the upper body. A Kinect camera, which is used for navigation as well as people and object detection and recognition, is placed on top of the robot using a pan and tilt mechanism.

\(^1\) http://www.roboticopenplatform.org/
2.2 SERGIO

Although AMIGO has performed well in RoboCup events and other projects, a number of possible improvements has been identified. Based on these issues a new robot, SERGIO (Second Edition Robot for Generic Indoor Operations), has been developed, which can be seen in Figure 2.

One the most important changes with respect to AMIGO is the wheel suspension. Due to the stiff design of AMIGO, the wheels of the robot easily lose traction when driving on a rigid, (slightly) uneven floor. Therefore, an advanced multi-link suspension has been designed for SERGIO such that the base is compliant for uneven floors, while keeping the wheels perpendicular to the floor at all times to avoid vibrations.

The torso consists of two degrees of freedom. A nearly translational vertical movement is combined with a rotating hip tilt movement. The translational degree of freedom is formed by a four bar straight guide mechanism, which moves the hip up and down. The hip tilt mechanism is helpful in two ways: it reduces the required vertical stroke and it enables the robot to bend forward to reach for objects further away. Both degrees of freedom are actuated using ball screw spindle mechanisms, while gas springs are used to reduce the required actuator forces.

A further key feature of the robot is its modularity. The design of SERGIO consists of four main modules: base, torso and two arms. Four M6 bolts in a square of 250 mm attach the torso to the base platform, while the arms slide in a simple tube with inner diameter of 140 mm. Electrically, there is a 24 V power connection, EtherCAT bus for data communication and a safety circuit between the various parts. For communication between the computers in the base and the torso, an additional ethernet connection between these modules is present. By designing modular robots, the exchangeability of components is stimulated so that modules of different manufacturers can be easily combined into a single robot. Furthermore, modularity eases maintenance of the robot.

2.3 The Tech United Eindhoven Software

The software on the robots is developed within ROS [11] and Orocos [2]. ROS provides a structured communication layer on top of the host operating system of a computer cluster and has a large suite of user-contributed packages and stacks for various functions, e.g., planning, navigation, perception and simulation. The software is partly based on these packages, which have been extensively modified. To control the hardware, the Orocos Real-Time Toolkit (RTT) is used, which is developed to build highly configurable and interactive component-based real-time control applications.
3 Navigation

This section describes the navigation system that is used by the AMIGO robot. AMIGOs navigation can be initialized with or without an a priori, static map. When initialized without a static map, a 2D localization map and 3D obstacle map are acquired simultaneously with localization using openSLAM Gmapping software. When using a map, AMIGO localizes itself using an Adaptive Monte Carlo Localization (AMCL) algorithm.

3.1 3D environment representation based on probability of collision

A 3D representation has been implemented last year, as described in [8], using the octree-based volumetric OctoMap framework [6]. This representation probabilistically fuses sensor measurements to model the occupancy of voxels as free, occupied or unknown.

However, during navigation a robot faces uncertainty in its estimated position and uncertainty due to external disturbances acting on the robot. To enable safer robot navigation, an integrated approach to deal with the multiple sources of uncertainty that are present in a domestic environment has been developed over the past year. The general idea of this approach is to determine a safe velocity limit for the robot based on the probability of collision. This approach consists of four steps:

1. Determine the probability of a voxel being occupied by an object. The existing OctoMap representation has been extended with a model that also describes the occupancy probability if no measurements are received, as shown in Figure 3. This avoids over-confidence in the assumption that voxels remain free if not observed. To keep the computational complexity of the planning tractable, the 3D representation is projected down to a 2D grid, as depicted in Figure 4. Unknown cells in this 2D grid map are inflated to deal with the possibility of obstacles appearing on the robot’s path from an occluded part of the environment. This is modeled using the pro-active approach [1], as shown in Figure 5.

2. Determine the probability of a voxel being occupied by the robot. The robot position uncertainty is modeled as a bivariate normal (Gaussian) distribution \( \mathbf{x} \sim \mathcal{N}_2(\mu, \Sigma) \), where the mean \( \mu \) is the robot position at coordinate \( \mathbf{x} = (x, y)^T \) and \( \Sigma \) is the two-dimensional covariance matrix. The occupancy probability of a cell \( c \) by the robot can be determined according to the normalized distribution \( \mathcal{N}(c; \mu, \Sigma) r^2 \), where \( r^2 \) is the surface of \( c \).

3. Determine the probability of collision by combining the probability of a cell being occupied by both an obstacle as well as the robot.

4. Relate the probability of collision to a safe velocity limit, depending on how dynamic the environment is.

![Figure 3](image.png)

**Fig. 3.** The occupancy probability of a voxel \( n_i \) if measurements are received and if no measurements are received for free voxels (left) and for occupied voxels (occupied).
3.2 Global planner and local reactive algorithm

The 3D environment representation, introduced in Section 3.1, is used in a motion planning approach with a global planner and a local reactive algorithm. The global A* planner searches for a plan in the form of a sequence of positions \((x, y)\) to the goal and the local reactive algorithm computes a velocity such that this global path is followed while attaining the velocity limit that the representation imposes. Re-planning is performed at a fixed frequency of 2 Hz. A re-plan is executed if it is significantly better in terms of the cumulative cost of the path. High-level decisions are made in an executive layer on top of the planners using a state machine. These decisions include recovery behavior, e.g., clearing a part of the map if AMIGO thinks it is stuck, and the gaze direction of the Kinect at a fixed distance on the global path in front of AMIGO.

4 Whole-body framework

In this section the transition towards a framework where the joint motions are generated simultaneously for all joints and while obeying several (motion) objectives is discussed.

4.1 Whole-body controller

A complete framework to handle several (motion) objectives simultaneously in a dynamic environment is proposed by [4], to construct safe and predictable motions for operations alongside humans using a large number of DoF. A scheme of the whole-body control framework is depicted in Figure 6. Task execution is solved by a Cartesian impedance controller, which spans a virtual linear spring between the current and desired end-effector pose. The collision avoidance module works in a similar way and prevents the robotic system from colliding with itself or its surroundings at all times. The joint torques \(\tau_{\text{imp,coll}}\) are computed by premultiplying the generated wrenches \(W\) with the transpose of the relevant Jacobian matrix \(J\): \(\tau_{\text{imp,coll}} = J^TW\). Staying away from the mechanical joint limits and realizing a desired posture are secondary motion objectives, operating in the null space of the Cartesian impedance and collision avoidance. The generated joint torques \(\tau_d\) resulting from the various motion objectives are sent to an admittance controller, which maps the joint torques \(\tau_d\) to desired joint positions \(q_d\).

The whole-body controller works well in dynamic environments for robotic systems with a large number of DoF. However, as the motion objectives of the whole-body control framework are based on potential functions, the whole-body controller is also prone to become trapped in local minima when operating in complex environments because it has no knowledge on the global connectivity of the environment [12]. A path planner which captures the global connectivity is developed to construct a path or a sequence of end-effector poses to keep away or recover from local minima which will be discussed next.
Fig. 6. A control architecture for reactive whole-body control with the primary motion objectives of task execution and safety. The secondary motion objectives are avoiding the mechanical endstops of the joints and realizing a desired posture.

4.2 Whole-body planner

The global connectivity of the workspace is represented by a visibility-based probabilistic roadmap [7]. This representation is searched by the $A^*$ search algorithm followed by a shortcutting and smoothing step to improve the end-effector path quality.

Since only the end-effector is considered during the construction of the stochastic representation and the path searching, a joint-space validation of the end-effector path is imperative to ensure the path is realizable for the complete robot. The joint motions are validated by integrating the whole-body controller forward, without executing it on the real robot. A simplistic dynamic model of relevant joints is identified for the purpose of accurate imitation of the real robot during the forward integration.

5 Perception

This section describes the perceptual capabilities of AMIGO. First, the object detection and recognition framework is explained in Section 5.1. Then Section 5.2 explains how people are detected and recognized.

5.1 Object detection and recognition

During the past years, various algorithms were used for object recognition. Experience has shown that a single algorithm is rarely sufficient to handle the wide variety of objects encountered in domestic environments. For that reason, a combination of algorithms is used. The result is a pipeline containing a sequence of different algorithms.

Under normal operation, the first component is an object segmentation module. Objects on top of a horizontal plane are segmented using the point cloud library (PCL)\(^2\). This module also approximates the size of the segmented objects and based on the size selects a subset of candidate objects from the object database. In Figure 7(a) the plane is represented by white points, whereas the object clusters are shown in pink.

The segmented objects and the related color image regions are given to the next module which contains a custom implementation of the linemod algorithm [3], see Figure 7(b). This algorithm recognizes objects based on 2D and 3D gradient templates and color information. The module refines the subset of candidate objects provided by the segmentation module.

The last module is based on the objects of daily use finder perception system [10], which is implemented by the ROS ODUfinder package\(^3\). This system aims at recognizing textures objects by adopting SIFT features using vocabulary trees, see Figure 7(c).

\(^2\) http://pointclouds.org

\(^3\) http://wiki.ros.org/objects_of_daily_use_finder
Fig. 7. White points in (a) represent the plane, pink the segmented objects. In (b), the camera image, templates, the depth image and similarity scores are shown (left to right and top to bottom). In (c) typical SIFT points used for object recognition and in (d), laser data used for estimating legs (green) and torsos (red).

The object detection and recognition system has a probabilistic nature. Each of the candidate objects given to the next module is associated with a probability. The module refines the probabilities and removes unlikely object candidates. The last module in the pipeline provides the object position together with a probability mass function over the possible class labels to the world model. If none of the objects is associated with a sufficient probability, the world model will insert an ‘unknown’ object at the given location.

5.2 People detection and recognition

Compared to objects, people are typically positioned further away from the robot. For that reason, AMIGO uses laser range finders for detecting people. Typical input data is shown in Figure 7(d).

The laser on AMIGO’s base is able to detect legs, whereas the higher laser can be used for detecting human torsos. Both the leg and torso detection are done by first performing a basic segmentation step based on the distance between subsequent laser points. Then, the classifiers proposed in [9] are used for the classification of segments. Finally the leg and torso information is merged, reducing the number of false detections.

For face detection and recognition, standard tools implemented in OpenCV\textsuperscript{4} are used.

6 World model

The tasks a service robot needs to execute in a household environment are typically object-centered. Therefore, the robot needs a representation of these objects and their relative positions,

\textsuperscript{4}http://opencv.org
and needs the ability to update this representation to account for changes in the environment. Such a representation will here be referred to as world model.

6.1 WIRE

The AMIGO robot uses the WIRE world model which was released early 2013 [5]. WIRE (World Informer for Robot Environments) is a probabilistic framework which uses Bayesian filters and object motion models to track objects and deal with sensor noise. Furthermore, it uses a multiple hypothesis filter to associate object measurements with existing objects in the world model. WIRE’s input consists of object measurements, or object evidence, generated by the perception routines described in Section 5. These measurements contain a semantic label describing the type of object, the position of the object, and possibly other attributes such as color. The WIRE module was successfully used during the RoboCup 2013, where its capabilities were especially shown during the Follow Me challenge.

WIRE was further developed to better deal with large numbers of object measurements from different sources and measurements that do not arrive ordered in time by introducing an evidence buffer and a world model history. If the world model is currently occupied with processing a detection, the received input is stored in the buffer. If then the stored measurement is found to be older, the world model is reverted back to a previous state and both measurements are fused in the correct order. Although this method introduces a slight overhead in computational complexity, we found it to be more reliable and robust in practice.

6.2 Towards a volumetric world model

The WIRE module represents objects as points in space, i.e., no volumetric data is associated with the objects. Furthermore, the module heavily relies on the quality of the perception routines: only objects that can be detected by the perception system can be represented and updated in the world model. In practice this means that only small-sized objects and persons are accounted for in AMIGO’s world representation, and therefore the navigation and manipulation modules need a different, typically grid-based geometric world representation for planning and collision avoidance. Our aim is to merge both representations into one, global, volumetric world view that can be used for navigation and manipulation and can answer the semantic object queries WIRE can already answer.

To this end, we started implementing a new world modeling system which represents objects as geometric shapes. Currently, primitive shapes, meshes and octrees [6] are supported. Object poses are stored as transformations between object frames. This means that objects poses are not necessarily stored in one common, absolute frame, but can also be represented with respect to each other. Each transformation is stored as a set of weighted particles such that pose uncertainty can be dealt with.

We assume that large, static parts of the environment - walls, fixed cabinets and such - are fed into the world model as prior information. This information is then used for the robot’s localization using standard methods such as AMCL, e.g., by projecting the objects down to a 2D occupancy grid. The known part of the world model is then rendered as if the sensors where observing it: objects are rasterized to a depth image to produce data similar to a Kinect image, and plane intersections are calculated to simulate LRF readings. Then, true sensor data and rendered data are compared and used to update the world model as follows: 1) rendered static objects are fitted into the measured data to improve the estimated sensor pose; 2) possibly moving objects are fitted into the data to update their position; 3) unexplained objects are removed from the world model; 4) unexplained sensor data is clustered and added as unknown object to the world model. Perception routines can then be used to try to identify and semantically label the unidentified, segmented objects.
6.3 Avoiding Hard-coded Knowledge

The above-mentioned method allows us to deal with changes in the environment, although certain objects, e.g., furniture, are not recognized by perception modules. This opens up a whole new range of possibilities. In previous years, robot waypoints, regions of interest and drop-off locations where hard-coded as Cartesian coordinates in an absolute frame, determined by the location of larger objects in the room. Once the world model can be used to determine and update the position of these larger objects, the waypoints, regions of interest and other interesting locations can be sampled with respect to the true poses of these objects during the execution of the task. For example, the robot can still find a suitable pose in front of the table to look for and grasp objects, even if this table has moved w.r.t. its original pose.

7 Conclusions

In this paper, this year’s main developments of Tech United Eindhoven have been discussed:

- a new robot has been developed, which has a compliant wheel suspension, increased workspace w.r.t. AMIGO is modular.
- 3D navigation with explicit uncertainty modeling results in safe navigation in a domestic environment.
- a whole-body planner and controller result in predictable, safe manipulation.
- the addition of SIFT object recognition to our perception stack improves object recognition, while combining the data of leg detections and torso detections leads to increased robustness in people detection.
- adding volumetric information to the world model minimizes the amount of hard-coded knowledge that is required to complete a RoboCup challenge.

With these improvements, we hope to improve on last year’s performance and are looking forward to RoboCup 2014 in João Pessoa!

References