Skuba 2012 Team Description

Kanjanapan Sukvichai[1], Teeratath Ariyachartphadungkit[1],Bhirawich Pholpoke[1], Khakana Timachai[2], Nuttapol Runsewa[2] and Sagpichest Sarinyavajn[3]

[1] Dept. of Electrical Engineering, Faculty of Engineering, Kasetsart University.
[2] Dept. of Computer Engineering, Faculty of Engineering, Kasetsart University.
[3] Dept. of Mechanical Engineering, Faculty of Engineering, Kasetsart University.

50 Ngarmwongwan Rd, Ladyao Jatujak, Bangkok, 10900, Thailand
skuba2002@gmail.com
http://iml.cpe.ku.ac.th/skuba

Abstract. This paper is used to describe the Skuba @home League robot team. Skuba robot is designed under the World RoboCup 2012 rules in order to participate in the @home competition in Thailand and in RoboCup. The overview describes both the robot hardware and the overall software architecture of our team.

Keywords: @home, Robocup, Manipulation, Robot Control, Face recognition, Voice recognition, People Tracking, SURF, NNN.

1 Introduction

Skuba is a new @home robot team from Kasetsart University [1]. We first entered the Thailand Robot@home 2011 competition and passed through the finalist.
Our robot is designed in order to take care of senior citizens. Its low level is based on Skuba small-size robot version 2009. Omni-directional wheels robot is one of the most popular mobile robot which is used in most of the teams because of its maneuverability. The new one robot has been creating in this year to solve many problems that appear in our Thailand robot @home 2011 competition.

2 Robot

The Skuba @Home robotic platform **Fig. 1** is based on the Skuba's robotic soccer platform with some differences. The robot has a Hexagon base made from Aluminum number 6061 and its size is 3mm thick with radius of 340 mm and height of 700 mm. Maxon EC 90 flat (brushless motor, 90 watt) with Planetary gear head (ratio 19:1) is selected to be a robot wheel driver and Robotis Dynamixel RX-28 (37.7 kg-cm holding torque) and RX-64 (64 kg-cm holding torque) is used as a manipulator driver.

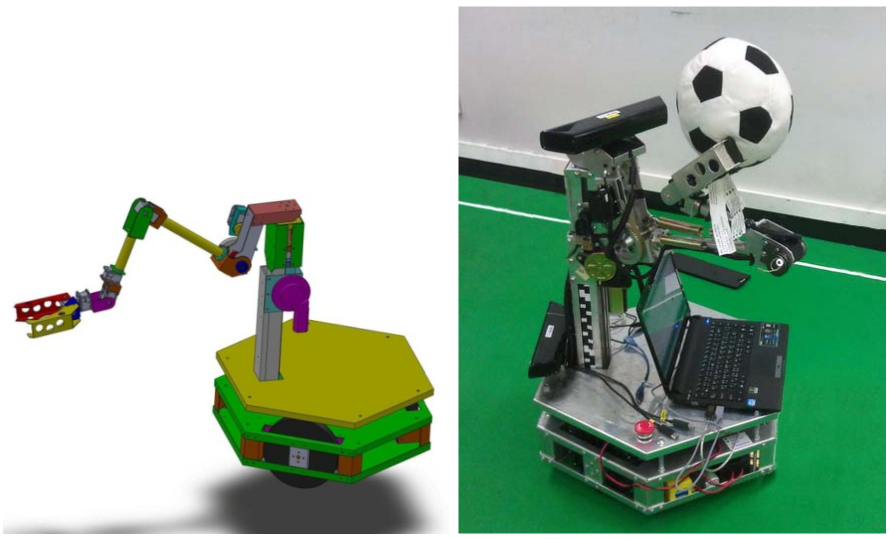

Fig. 1. 3D mechanical model of Skuba 2011 robot@home

We can separately describe our robot into two layers, that is, the high level layer and the low level layer.

2.1 The high level layer

The high level layer consist of two Microsoft Kinect sensor (one is fixed on the top and another is fixed near the floor) as our vision system and laptop computer as a processing unit.

Microsoft Kinect provides real time RGB and depth image to each module **Fig. 2** in the processing unit.

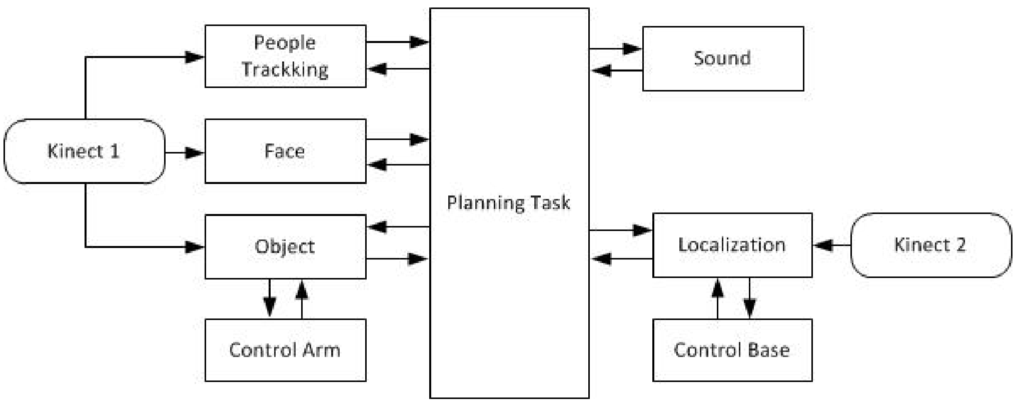

Fig. 2. SKUBA Software Diagram

2.1.1 Software Architecture

Our robot is based on the ROS (Robot Operating System), which is commonly used in many robot applications. In our design, we divide the processing unit into many separated modules which each module communicates by using ROS libraries. The advantage of being separated modules is to be convenient to work with.

2.1.1.1 Speech Synthesis

The e-speak. It uses a "formant synthesis" method. This allows many languages to be provided in a small size. The speech is clear, and can be used at high speeds, but it is not as natural or smooth as larger synthesizers which are based on human speech recordings.

2.1.1.2 Speech Recognition

The pocketsphinx base on CMUSphinx speech recognizer . It uses gstreamer to automatically split the incoming audio into utterances to be recognized . Currently, the recognizer requires a language model and dictionary file. These can be automatically built from a corpus of sentences using the Online Sphinx Knowledge Base Tool.

2.1.1.3 Face Detection & Face Recognition

The Haar like Features Cascade is used to detect faces. Moreover, we make use of depth data from Kinect sensor to cut background. In face recognition part, we choose eigenface as our algorithm.

2.1.1.4 Object Recognition & manipulation

Development of the SURF(Speeded Up Robust Features) algorithm and the FLANN (Fast approx nearest neighbor classification) which is a Libraries of OpenCV.

The x,y of the object from a picture will be combined with depth data for commanding the manipulator in xyz coordination.

2.1.1.5 People tracking

We use Kinect sensor to get depth image for finding human body. The NNN (Naive Nearest Neighbors) was used. This library provides a naive way of doing a radius search for nearest neighbors. This can improve in tracking human in moving frame and human walking through obstacle. The robot velocity command is according to the distance and the angle between robot and human body.

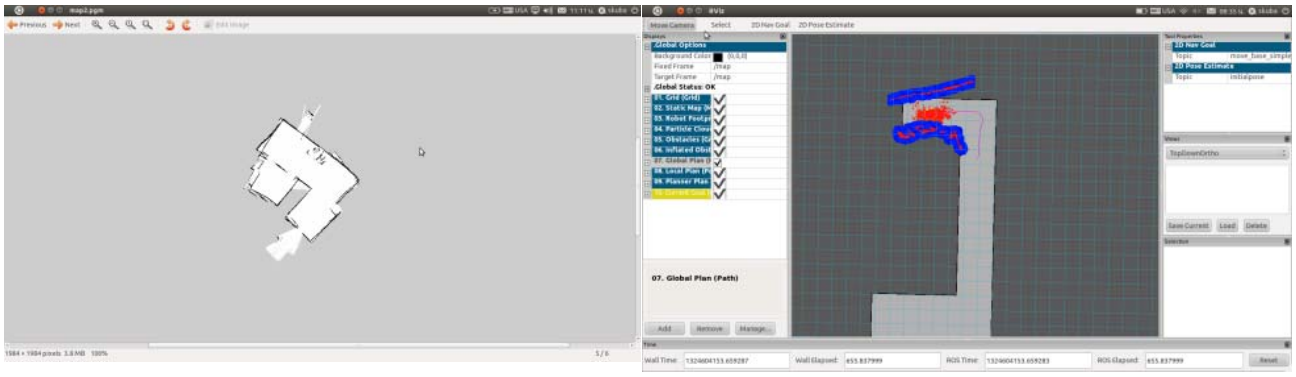

Fig. 3. OpenSLAM Gmapping

2.1.1.6 Localization and Navigation

The robot relies on a Kinect sensor to acquire environment data. We extract Kinect’s depth image and use it as a fake laser scanner. Additionally, odometry information is considered.

AMCL (Adaptive Monte Carlo Localiztion) system is used for robot localization. AMCL is a probabilistic localization system for a robot moving in 2D. It implements the adaptive (or KLD-sampling) Monte Carlo localization approach (as described by Dieter Fox), which uses a particle filter to track the pose of a robot against a known map.

This known map is acquired with OpenSLAM Gmapping Package **Fig.3**,which is a highly efficient Rao-Blackwellized particle filer to learn grid maps from laser range data.

ROS navigation stack is used for robot navigation. A 2D navigation stack that takes in information from odometry, sensor streams, and a goal pose commands safe velocity to a mobile base.

2.2 The low level layer

The low level layer is composed of motor control module, manipulator control module and sensor fusion module.

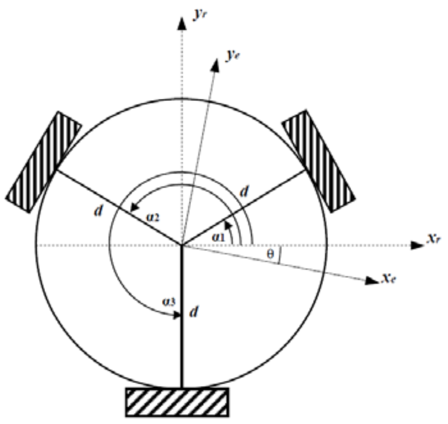

Fig. 4. Robot Dynamics

2.2.1 Motor control module

Motor control module is based on Spartan 3 FPGA from Xilinx. The FPGA contains a soft 32-bit microprocessor core and peripherals. This embedded processor executes the low level motor control loop, communication and debugging. The motor controller, quadrature decoder, PWM generation and onboard serial interfaces are implemented using FPGA.

The PI controller is employed to achieve each wheel velocity. Each wheel velocity is converted from the given velocity with respect to earth frame that can be written as:

$$\zeta_{wheel} = \psi \cdot \zeta_{earth} \qquad (1)$$

$\zeta_{wheel} = [\dot{\phi}_1 \quad \dot{\phi}_2 \quad \dot{\phi}_3]^T$

$\zeta_{earth} = [\dot{x} \quad \dot{y} \quad \dot{\theta}]^T$

$$\psi = \begin{bmatrix} cos\theta \cdot sin\alpha_1 + cos\alpha_1 \cdot sin\theta & sin\theta \cdot sin\alpha_1 - cos\alpha_1 \cdot cos\theta & -d \\ cos\theta \cdot sin\alpha_2 + cos\alpha_2 \cdot sin\theta & sin\theta \cdot sin\alpha_2 - cos\alpha_2 \cdot cos\theta & -d \\ cos\theta \cdot sin\alpha_3 + cos\alpha_3 \cdot sin\theta & sin\theta \cdot sin\alpha_3 - cos\alpha_3 \cdot cos\theta & -d \end{bmatrix}$$

From (1) and **Fig. 4**

ψ	is the kinematic equation
α_i	is the angle between wheel *i* and the robot x-axis
$\ddot{\theta}$	is the robot angular acceleration about the z-axis of the global reference frame
ϕ_i	is the angular velocity of wheel *i*
d	is the distance between wheels and the robot center

2.2.2 Manipulator control module

The manipulator **(Fig. 5)** is controlled by using differential inverse kinematics processed by an ARM Cortex M3 Microprocessor to follow a fixed trajectory with its 5 degree of freedoms. The fixed trajectory is selected by considering the paths that are possible to have no obstacles.

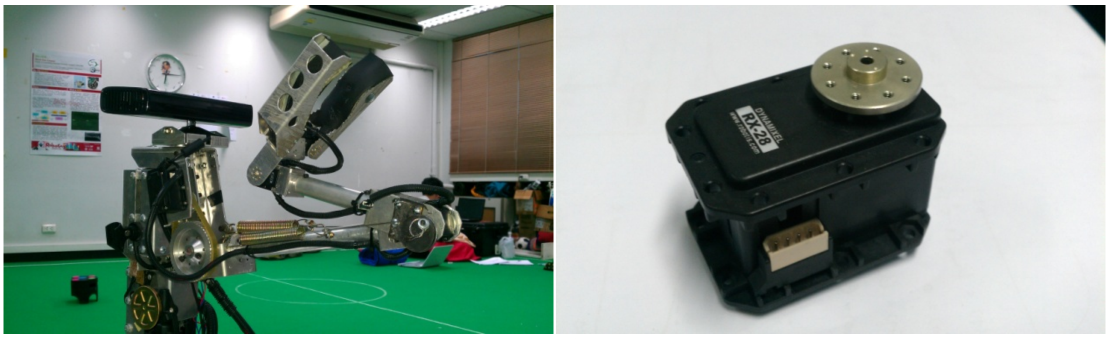

Fig. 5. Manipulator and Robotis Dynamixel servos

The geometric approach[7] is selected to solve inverse kinematic equations. We choose P2 in **Fig.6** as a decoupling point.

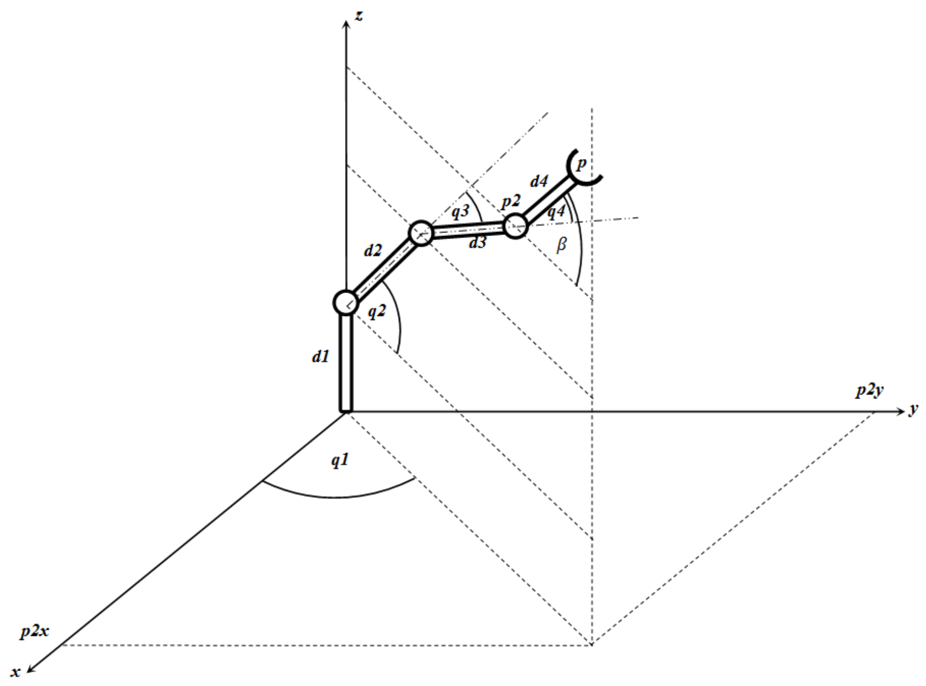

Fig. 6. Kinematics of manipulator

$$q_1 = arctan2(p_y, p_x)$$
$$p_2x = p_x - d_4 \cdot \cos(\beta) \cdot \cos(q_1)$$
$$p_2y = p_y - d_4 \cdot \cos(\beta) \cdot \sin(q_1)$$
$$p_2z = p_z - d_4 \cdot \sin(\beta)$$

And the distance *d* from joint 2 to point P2 is

$$d^2 = p_2x^2 + p_2y^2 + (p_2z - d_1)^2 \qquad (2)$$
$$d^2 = {d_2}^2 + {d_3}^2 - 2 \cdot d_2 \cdot d_3 \cdot \cos(\pi + q_3) \qquad (3)$$

We can solve q_3 from (2) and (3) by

$$q_3 = arctan2\left(-\sqrt{1 - \cos(q_3)^2}, \cos(q_3)\right)$$

Next, we can use q_3 to find q_2

$$q_2 = arctan2\left(p_2z - d1, \sqrt{p_2x^2 + p_2y^2}\right) - arctan2(d3 \cdot \sin(q_3), d2 + d3 \cdot \cos(q_3))$$

And

$$q4 = \beta - (q_2 + q_3)$$

The IR distance sensor confirms the destination point provided directly from the high level layer to send feedback whether the object is reach or not.

2.2.3 Sensor fusion module

This module based on another ARM Cortex M3 Microprocessor receives data from FPGA, which consists of velocity in X, Y and angular velocity from wheel encoders, and data from yaw-axis gyroscope **(Fig.7)**. This helps to minimize the mean square error of the non-perfect sensor measurement originated from wheel slippage and gyro drift over time. We choose a kalman filter represented below as our solution.

$$x_t = F \cdot x_{t-1} + w_t \qquad (4)$$
$$z_t = H \cdot x_t + v_t \qquad (5)$$

In (4) and (5), x_t is the state vector, *F* is the state transition matrix, z_t is the measurement vector, *H* is the measurement matrix, w_t and v_t is the process noise and measurement noise with zero mean respectively.

The state vector is given by

$$x = \begin{bmatrix} \theta \\ \omega \\ b \end{bmatrix}_t \text{ ; where } b \text{ is the gyro bias.}$$

The state transition matrix is given by

$$F = \begin{bmatrix} 1 & \Delta T & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \text{ ; where } \Delta T \text{ is the time interval.}$$

The measurement vector is given by

$$z_t = \begin{bmatrix} \omega_{wheel} \\ \omega_{gyro} \end{bmatrix}$$

And the measurement matrix is given by

$$H = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix}$$

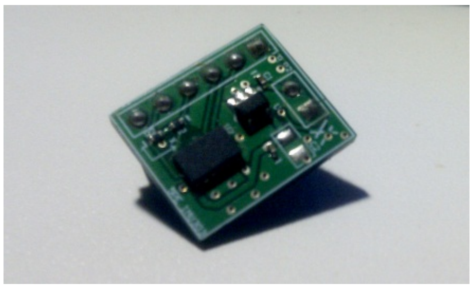
Fig. 7. 9DOF IMU (Inertial Measurement Unit)

Besides, this module communicates directly between the high level layer and motor control module: the data sent to the high level layer is an odometery and the data sent to the motor control module is a desired velocity.

3. Conclusion

In this year, we mainly focus on developing a stable robot and also its AI software. We have experiences from Thailand robot @Home competition 2011 and we learn from our mistakes. Our robot and AI are improved in order to join RoboCup 2012 event in Mexico. We hope that our robot team will perform better in RoboCup than in Thailand, and we are looking forward to sharing experiences with other great teams around the world.

References

1. Srisabye, J., Hoonsuwan, P., Bowarnkitiwong, S., Onman, C., Wasuntapichaikul, P., Signhakarn, A., et al., Skuba 2008 Team Description of the World RobCup 2008, Kasetsart University, Thailand.
2. Oliveira, H., Sousa, A., Moreira, A., Casto, P., Precise Modeling of a Four Wheeled Omni-directional Robot, Proc. Robotica'2008 (pp. 57-62), 2008.
3. Rojas, R., Forster, A., Holonomic Control of a robot with an omnidirectional drive, Kunstliche Intelligenz, BottcherIT Verlag, 2006.
4. Maxon motor, Key information on – maxon DC motor and maxon EC, Maxon Motor Catalogue 07(pp. 199), 2007.
5. http://espeak.sourceforge.net/
6. http://cmusphinx.sourceforge.net/
7. Krzysztof Tokarz, Slawosz Kieltyka, Geometric approach to inverse kinematics for arm manipulator, Institute of Informatics Silesian University of Technology, Poland