Thesis Interim Progress Report

Sotnichenko Roman
Innopolis University

2016

Table of Contents

1 Goals of the project

1.1 Objectives

The short-term aim of my project is to implement a remote control software for Android device to control DARwIn-OP2 (Dynamic Anthropomorphic Robot with Intelligence – Open Platform[1]). As a result we can send commands to robot by means of Wi-Fi or Bluetooth to execute different kinds of actions: step/walk forwards/backwards/left/right, sit down, stand up, rotate around it's axis, throw jabs/punches/hooks/crosses by left/right hand, make thrusts by both hands, throw direct/side/low/middle/high kicks, evade/block attacks.

The long-term aim is to automate this process so that two DARwIns can fight each other with minimum human interaction, using such robotics forums like computer vision and stabilization.

1.2 Report's overview

- Overview of the system specification: in this chapter I provide an overview of the initial specification, setting out the main features of the system or project and addressing system functionality, interfaces, and signal/information representations;

- Background theory: here I describe the essential theory to be applied in the thesis;

- Overview of task specification and project schedule: in the fourth chapter of the report I list the primary tasks and sub-tasks required to carry out the project and an overview of the project schedule, giving the timings (start date, duration, amount of effort) of each task;

- Review of tasks: in this chapter I give a comprehensive review of the status of each task and sub-task (setting out the status , problems encountered and identified solutions, anticipated problems and possible solutions, impact on the project schedule);

- Interim results: here I provide examples of interim results I have achieved;

- Short term plans: the last chapter identifies the next steps I will take in the project.

2 Overview of the system specification

2.1 The main features of DARwIn

As I mentioned in the first chapter, I use DARwIn-OP2 as it is an open platform robot. DARwIn-OP's main purpose is for research and programmers in the fields of humanoid, artificial intelligence, gait algorithm, vision, inverse kinematics, linguistics, etc... It is also supported by $1.2 million NSF grant[2][3] and has been distributed to over 14 institutions already. It has an advanced computational power, sophisticated sensors, high payload capacity, and dynamic motion ability developed and manufactured by Korean robot manufacturer ROBOTIS in collaboration with Virginia Tech, Purdue University, and University of Pennsylvania. DARwIn has twenty degrees of freedom (20 DOF).

2.2 Specifications

High performance and advanced features[4]:

- Default walking speed: 24.0 cm/sec (9.44 in/sec) 0.25 sec/step – user modifiable gait;

- Default standing up time from ground: 2.8 sec (from facing down) and 3.9 sec (from facing up) – user modifiable speed;

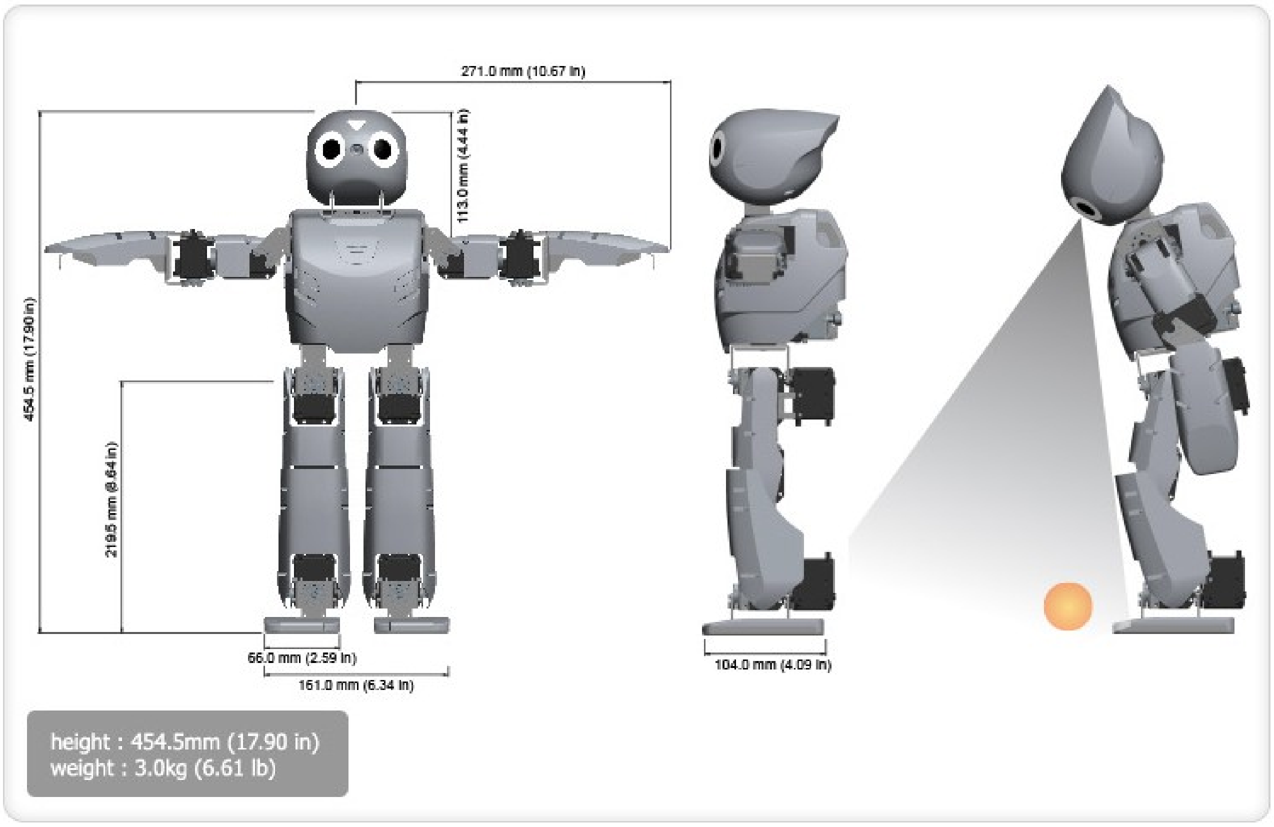

Figure 2.1 Dimensions and weight

- Built-in PC: Intel Atom N2600 @1.6 GHz dual core, RAM 4GB DDR3, 32GB mSATA;

- Management controller (CM-740): ARM CortexM3 STM32F103RE 72MHz;

- 20 actuator modules (6 DOF leg x2 + 3 DOF arm x2 + 2 DOF neck);

- Actuators with durable metallic gears (DYNAMIXEL MX-28T);

- Self-maintenance kit (easy to follow steps and instructions);

- Standby mode for low power consumption;

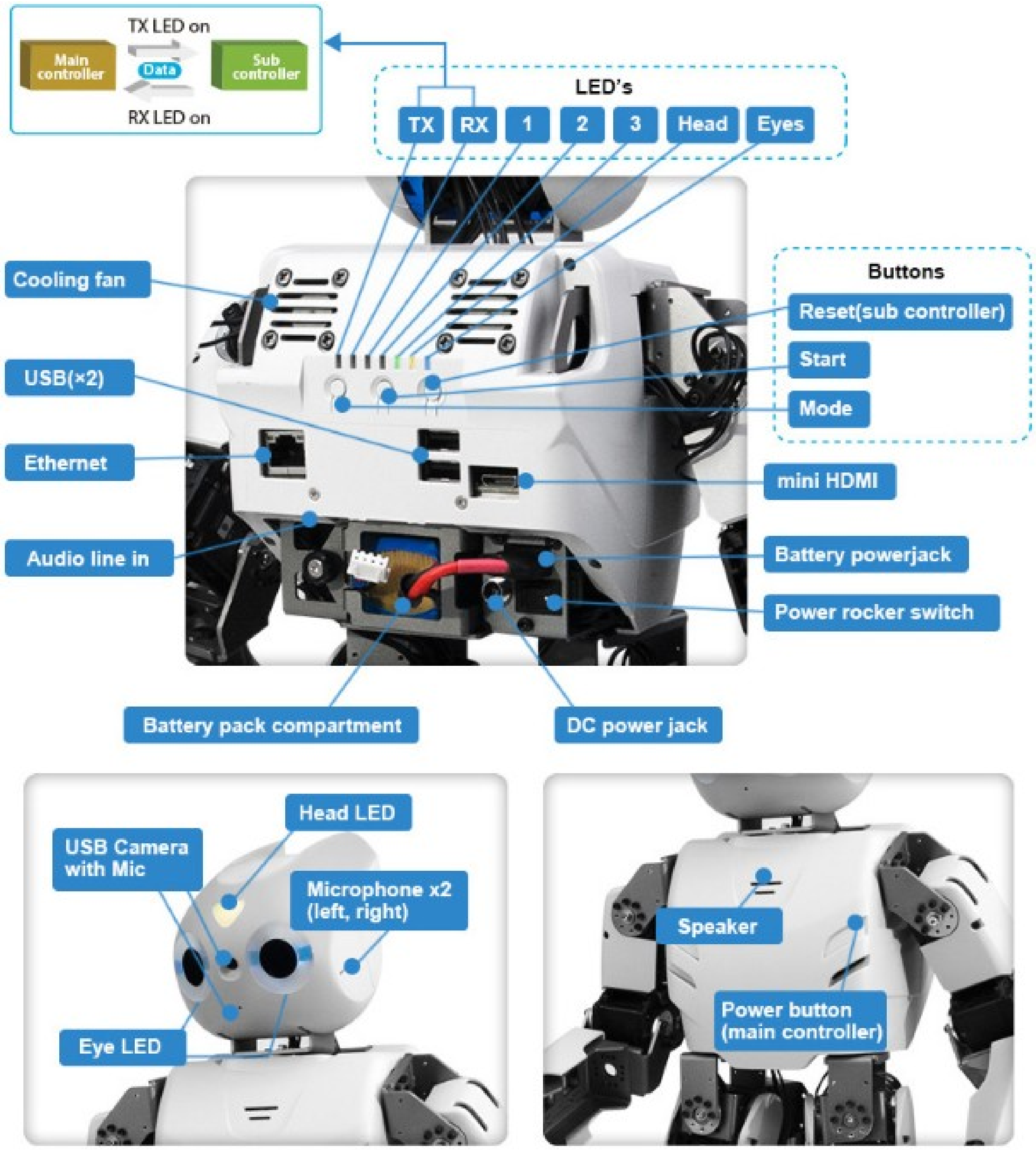

Figure 2.2 Standard PC-based robot with convenient interfaces

- 1Mbps high-speed Dynamixel bus for joint control;

- 1800mAh LIPO Battery (30 minutes of operations), charger, and external power adapter (Battery can be removed from robot without shutting down by plugging in external power before removal);

- Versatile functionality (can accept legacy, current, and future peripherals);

- 3-axis gyro, 3-axis accelerometer, button x3, detection microphone x2.

Open platform (Hardware and Software):

- Mechanics information (Dimensions, Kinematics, Dynamics, CAD data);

- Electronics information (Controllers, Sub B/Ds, Schematics, Part information);

- Software information (Development environment, Framework, Source code);

- Management information (Detailed assembly diagrams, User maintenance guide);

- Community resources (User-developed code, various application examples).

2.3 Software structure

The software aspect of DARwIn-OP has been built with a hierarchical framework considering modularity and independency[5]. The framework consists of device communication module, motion module, walking module, sensing module, behavior module, vision module, and diagnostics module. The framework has been developed with C++ programming language where the code is operating system-independent. The operating system-independent aspect of the framework is essential so that the code can be ported to any existing or future computer operating system, including the newly-developed Robot Operating System (ROS).

The user may simply write a behavioral code for DARwIn-OP without the need to develop a separate framework set. In such case software simulator is the most practical method of writing the program and testing said program for DArwIn-OP, given that such simulator makes use of the provided framework. Due to the open-source nature of DARwIn-OP the user is encouraged to share the developed program with other users. However, users may not be limited to open-DARwIn-SDK. There is currently other independently-developed software that does not implement open-DARwIn-SDK. There are some examples where software has been developed at “levels.” For instance low-level programming takes can take care of the robot’s sub-routines, such as camera refresh times or read the actuator’s position. The high-level programming can take care of the more abstract aspects of the robots. The different levels are practical so that recompiling of the entire code is unnecessary on simple changes in subroutine behaviors or parameters.

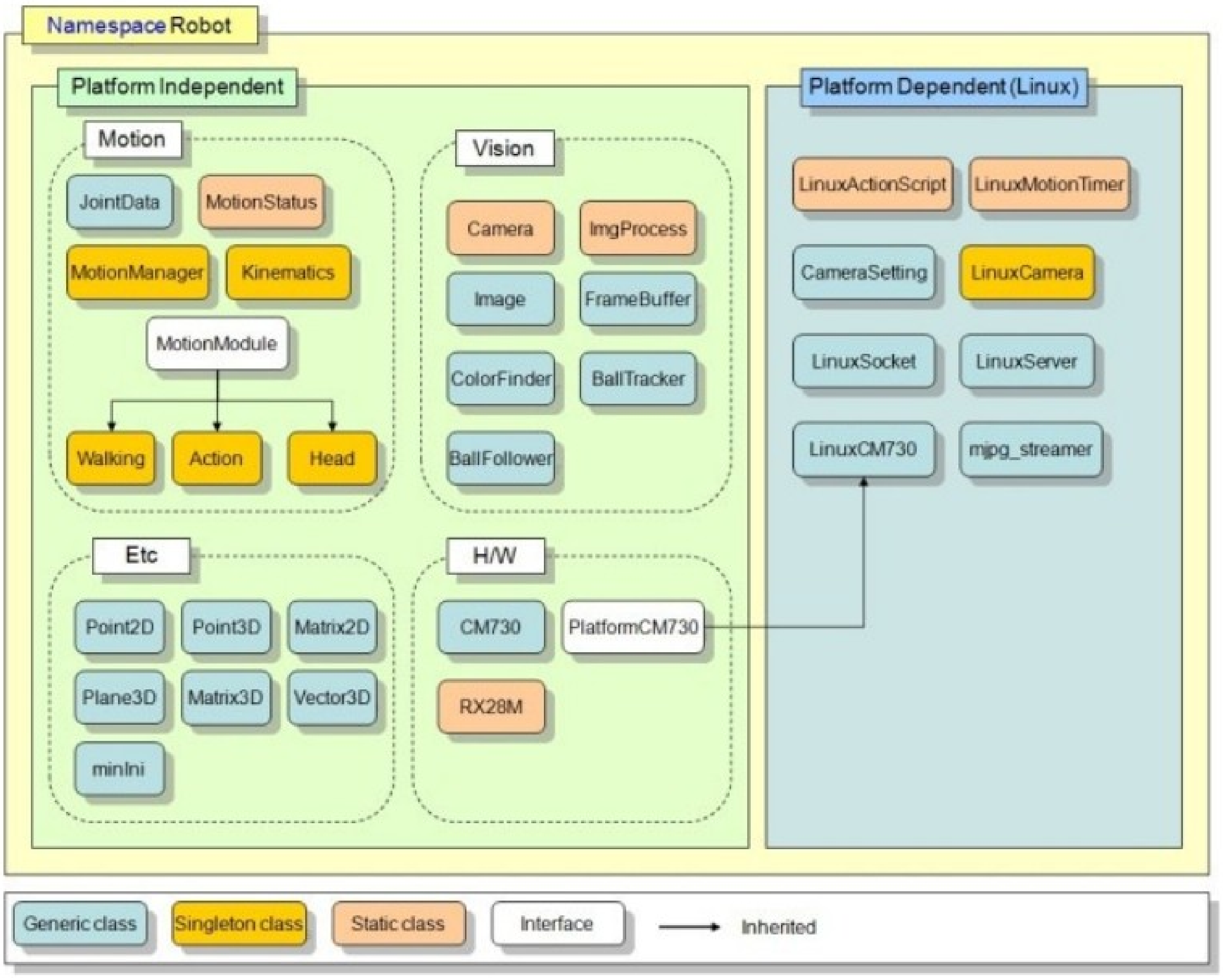

Figure 2.3 Software framework for DARwIn-OP

3 Background theory

3.1 Applications

3.1.1 Factory's algorithms

In factory's configuration for DARwIn-OP2 next algorithms are implemented:

- demonstration (greetings, parting, telling about it's abilities and features accompanied by gestures);

- playing football (DARwIn just looks for a small red ball by rotating his head, where camera is placed, and when he finds it, he gets closer to it, and then kicks the ball. Figure 3.1);

- walking.

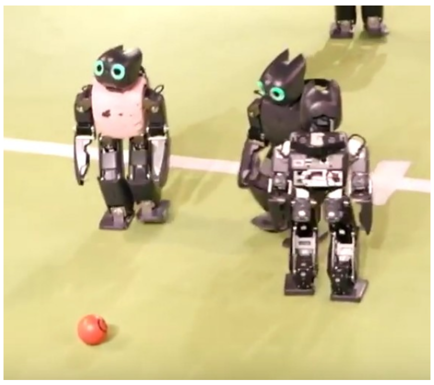
Figure 3.1

3.1.2 Children's autism

Autism Spectrum Disorder (ASD), a mental condition, is a neurobehavioral disorder that is present from early childhood and is characterized by impaired social interaction and difficulty in communicating and forming relationships with other people.

The purpose behind this initiative in correlation with autismis to enable robots to interact and help autistic children in communicating and socializing more effectively. Still in its initial phase, the robot project currently caters only to autistic children aged between 5 and 10 years[6]. Eventually, the project hopes to include even three-year-olds.

In autism, signs and characteristics of the disorder vary from one child to another. However, one thing common across the spectrum is that autistic children typically avoid eye contact. This hinders their ability to interact and communicate with family members, friends, and other people in general.

However, it was found that autistic children were more comfortable reaching out to robots because their actions can be controlled and were predictable, in comparison with humans.

In the current series of Park's experimentation, the researcher used three predominant types of robots - a mini robot, a medium robot, and a large robot such as the DARwIn-OP2. While the first two types of robots are capable of displaying facial emotions and physical actions respectively, the third category of robots is capable of engaging with children in a more interactive way such as playing sports even.

The artificial intelligence (AI) used by the robots enables them to analyze the behavior of a child. The acquired data is then assimilated to find different and appropriate ways to interact and reach out to the children with autism.

Experts in the field have welcomed the design, stating that the repetitive task of teaching social skills to an autistic child is the perfect skill a robot can diligently take over. These robots can also assist parents, who do not have enough time in terms of providing intensive therapy to the child.

3.1.3 Other applications

Also Computational Intelligence and Robotics Lab together with Department of Electrical Engineering and National Taiwan Normal University developed the next algorithms for overcoming obstacles for Darwin[7], showing the diversity of problems Darwin can solve:

- crawling under the barrier (figure 3.2);

- climbing the hand stairs (figure 3.3). The additional gripper arms module[8] (figure 3.4) was used;

- moving the rope hanging on it. Also with gripper arms (figure 3.5);

- going up and down the inclined surface (figure 3.6);

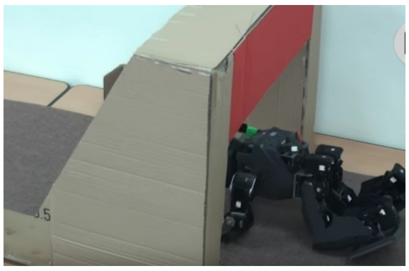
Figure 3.2

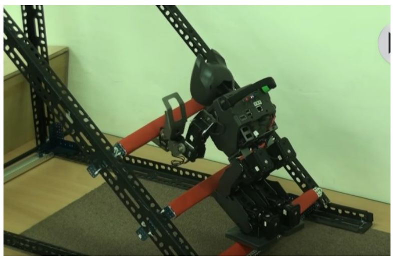
Figure 3.3

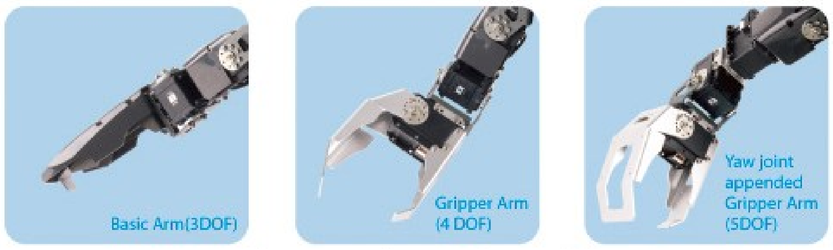

Figure 3.4

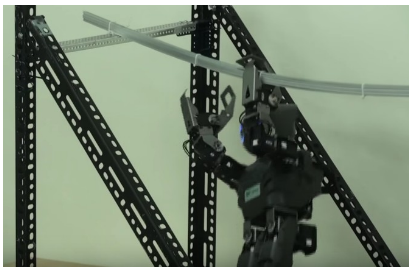
Figure 3.5

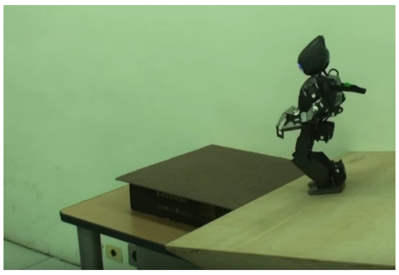
Figure 3.6

3.2 Issues

3.2.1 Stabilization

Many walking control techniques have been developed based on the assumption that the walking surface is perfectly flat with no inclination. Accordingly, most biped humanoid robots have performed dynamic walking on well designed flat floors. In reality, however, a typical room floor that appears to be flat has local and global inclinations of about 2 degrees. It is important to note that even slight unevenness of a floor can cause serious instability in biped walking robots.

Stability criteria

Different criteria exist in the literature. For static cases (when the robot does not move), there is a sufficient and necessary criterion: A posture is denoted static stable if and only if the projection of the center of mass on the ground (P CM) is within the convex hull of the support polygon.

But for dynamic cases (when the robot is in movement), the problem becomes much more complex. At current date there is still no sufficient and necessary criterion, but there are many necessary criteria for some assumptions. Here I briefly present the useful support points and a well-known criterion, namely the ZMP.

Walking pattern generation

For the design of the walking pattern, the authors[9] considered the following four design factors:

1) walking cycle (2 × step time);

2) lateral swing amplitude of the pelvis;

3) double support ratio;

4) forward landing position ratio of the pelvis.

The walking cycle was set by using the natural frequency of a 2D simple inverted pendulum model as shown in Fig.3.7 in order to produce natural swings. The natural frequency of the 2D simple inverted pendulum f_n is derived as equation (1). Here, l is the length of the pendulum and g is the gravitational acceleration.

$$f_n = 1/\pi \times \sqrt{g/l} \quad (1)$$

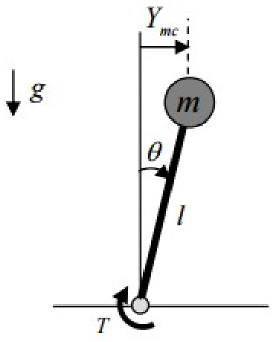

Figure 3.7

The lateral swing amplitude of the pelvis also can be derived by means of the ZMP dynamics of a 2D simple inverted pendulum model. The equation of motion of a simple inverted pendulum model can be written as follows:

$$T = mgl\Theta - ml^2\ddot{\Theta} \quad (2)$$

Where, T is the torque at the joint, m is the point mass and Θ is the angular displacement. Then, if we divide the right and left parts by mg, we have:

$$T/(mg) = l\Theta - (l/g) \times l\Theta \quad (3)$$

By substituting $F_z = mg$ and $l\Theta = Y_{mc}$ into equation (3) under an assumption that Θ is small (Θ < 5 degrees), then we obtain:

$$T/F_z = Y_{mc} - (l/g) \times \ddot{Y}_{mc} \quad (4)$$

Where, F_z is the ground reaction force, Y_{mc} is the lateral displacement of mass center. ZMP is practically calculated through dividing the torque T at the joint by the ground reaction force F_z . Thus, we can finally get following ZMP dynamics:

$$Y_{zmp} = Y_{mc} - (l/g) \times \ddot{Y}_{mc} \quad (5)$$

Where, Y_{zmp} is the lateral ZMP. When the robot is walking, we can assume the lateral displacement of the mass center as $Y_{mc} = A \sin\omega t$ on the coronal plane. Then, we obtain:

$$Y_{zmp} = A\left(1 + (l/g)\omega^2\right)\sin\omega t \quad (6)$$

Here, when the walking frequency $\omega(2\pi f_n)$ and the lateral swing amplitude of the pelvis A are chosen as 3.3 rad/sec and 32 mm, the amplitude of $Yzmp$ becomes 64 mm. Moreover, in real robot's walking, A becomes larger than the original value due to deflection of the compliant force/torque sensor structures attached at the ankle joints. If it is assumed that the robot tilts at about 0.5 degree more, the real A is about 39 mm, and hence $Yzmp$ becomes 78 mm. The $Yzmp$ of 78 mm is a suitable value for locating the ZMP on the near the foot center. In addition, its value has a roughly 10 percent of margin considering the lateral distance between the pelvis center and the ankle joint is 71 mm.

Walking control algorithm

The proposed walking control method is based on a controller switching strategy, and thus it is important to divide the walking cycle into several walking stages. In each walking stage, suitable online controllers are activated. Fig.3.8 shows the walking stages and their descriptions are as follows:

1) 1st stage : lift the left leg to its maximum flexion and height;

2) 2nd stage : lower the left leg until it makes complete contact with the ground;

3) 3rd stage : lift the right leg to its maximum flexion and height;

4) 4th stage : lower the right leg until it makes complete contact with the ground;

5) 5th stage : this stage follows the 1st or 3rd stage, and brings the robot to a standing pose with both legs landed on the ground.

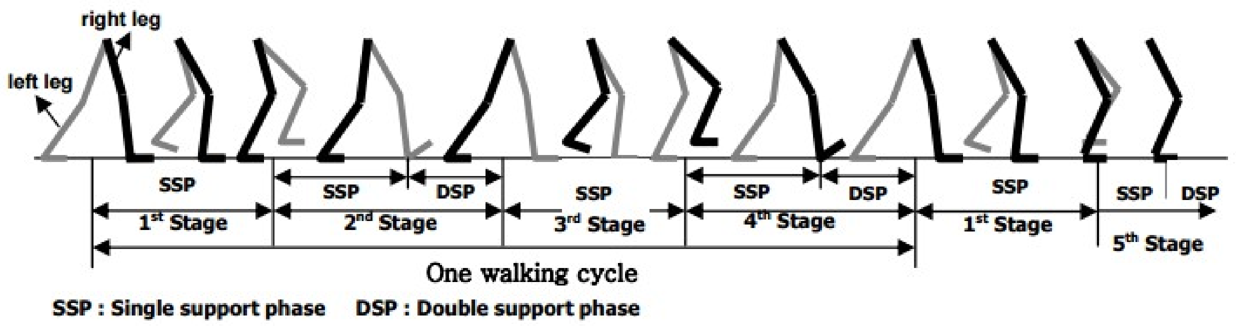

Figure 3.8 walking stage

During walking, the walking stages 1~4 are repeated continuously. In particular, in the 2nd and 4th walking stages, the single and double support phases coexist. The authors have developed a walking control algorithm that is composed of three control strategies[10], a real-time balance control strategy, a walking pattern control strategy, and a predicted motion control strategy. Each control strategy is composed of several online controllers according to the objective of the strategy. In Table 4, a previous walking control algorithm that does not consider uneven and inclined floor conditions is summarized. The previous walking control algorithm could make the robot walk stably on a normal room floor. However, the biped humanoid robots have to walk stably on uneven and inclined asphalt roads or footpaths in order to operate in the human society. Hence, online controllers that can cope with various ground conditions are integrated into the system. The proposed online controllers have been added to our previous walking control algorithm.

Table 3.1 Brief summary of the previous walking algorithm

Control scheme	Online controller (working period)	Objective
Real-time balance control	Damping controller (1 and 3 stages, SSPs of 2 and 4 stages)	Eliminate the upper body oscillations in single support phase by imposing damping at the ankle joints
	ZMP compensator (1 and 3 stages, SSPs of 2 and 4 stages)	Maintain dynamic balance by horizontal motions of the pelvis
	Soft landing controllers (DSPs of 2 and 4 Stages)	Absorb landing impact and adapt the foot to the ground surface
Walking pattern control	Pelvis swing amplitude controller (DSPs of 2 and 4 Stages)	Compensate the lateral swing amplitude of the pelvis by considering the amplitude of the ZMP
	Torso pitch/roll controller (DSPs of 2 and 4 Stages)	Compensate the center position of the pelvis swing to balance the pitch & roll inclinations of the torso
Predicted motion control	Tilt over controller (1 and 3 Stages)	Compensate the ankle joint trajectories to prevent “tilt over” in roll direction
	Landing position controller (2 and 4 Stages)	Compensate the landing position to prevent unstable landing

3.2.2 Computer vision

The core vision faculty includes seven researchers. Additions since 2005 include Yaser Sheikh, whose recruitment strengthened the areas related to 3D geometry and motion as well as computer graphics, and Fernando De la Torre, whose research addresses behavior and face analysis and is closely aligned with work in the machine learning group. Researchers address all of the main areas of computer vision: object recognition and scene analysis (Efros, Hebert, Kanade); 3D geometry and reconstruction (Kanade, Sheikh); motion analysis and tracking (De la Torre, Hebert, Kanade, Sheikh); physics-based vision (Narasimhan); and analysis of 3D data (Hebert, Huber).

The pace of progress in the areas of object recognition and scene analysis has increased over the past five years in the computer vision community, owing in part to more sophisticated use of the tools from machine learning and the availability of large data sets. The CMU group has greatly expanded its activities in these areas through close collaborations with the machine learning area, as documented in the Machine Learning section. In particular, the vision group is a leading contributor to the transformation of the field in the last five years from pattern classification of image pixels to deep understanding of scenes, including geometry, context, and other physical world constraints (Efros, Hebert, Kanade). This research area has considerable potential given the unique collaborations with researchers in Machine Learning and AI. Another important development in the last five years is the access to huge amounts of visual data (images and videos), for example from web repositories, which raises new research questions and offers new opportunities for computer vision algorithms. We (Efros) are at the forefront of this transformation through major contributions to defining new ways of addressing computer vision tasks that exploit the richness of the data sets.

At the other end of the spectrum, recent years have seen a resurgence of interest in recognition tasks in which the goal is to recognize instances of a specific object rather than broad categories of objects. Interest in this problem is based on renewed interest in classic robotics problems, such as bin-picking, but also in human-centric vision tasks in which one goal is to understand a person's environment. New opportunities in this area are developing, through industry funding (Kanade) and through the QoLT ERC (Hebert), respectively, and we expect this area to grow in the future.

In the past three decades, the RI vision group has made major contributions to the problem of reconstructing the 3D geometry of a scene from multiple views. While this problem is now well addressed in the case of static environments, 3D understanding of dynamic environments poses difficult challenges, which are being addressed (Kanade, Sheikh) through an ambitious research program. This includes investigating the use of a large number of imaging sensors based on the concept that complex sensing tasks, such as dynamic 3D scene understanding, should be solved by a large number of parallel but simple perceptual processes. As cameras proliferate in society with hand-held devices, this research will enable research into large-scale sensing challenges.

All aspects of human sensing, such as detecting, tracking, and understanding peoples' faces, bodies, and activities are addressed by the vision group (Cohn, De la Torre, Kanade, Lucey, Sheikh). This research has led to important developments both in the theoretical and the algorithmic aspects, including facial expression analysis (Cohn, De ,la Torre) and body posture recovery and tracking (Kanade, Sheikh). Work on recognizing actions and activities is expanding through new approaches for recognition in videos (Hebert, Sheikh). In addition to the research products, the vision group has had major impact on the field through several databases and benchmarks, e.g., the face databases and the Grand Challenge database of human activities generated in the QoLT ERC.

The area of physics-based vision is led since Fall 2004 by Narasimhan[11] who has developed it into three key areas: the mathematical modeling of the interactions of light with materials and the atmosphere; the design of novel cameras with higher resolution in space, color, and intensity; and the development of algorithms for rendering and interpreting scene appearance. This work contributed fundamental tools toward modeling and understanding light transport and reflection and it generated new applications in a number of fields including robotics, digital entertainment, remote sensing, and underwater imaging. Most recent work includes the development of new solutions for structured light sensing, and new display technology. This activity is central to the unique strength of RI in research activities combining computer vision and computer graphics.

As computer vision research matures, opportunities for applications and collaborations have increased. Accordingly, in addition to the basic research lines, the computer vision group collaborates with, and contributes to a large number of other areas and projects both within RI and with other parts of the University.

Video interpretation tasks, e.g., for surveillance, are traditional settings for tracking and motion analysis research. One important direction for this area is towards systems that incorporate higher level of reasoning. The vision faculty is well-positioned for this research direction, given the strong ties to colleagues in other parts of SCS. For example, the most recent effort in this area involves a collaboration between researchers in vision (Hebert, Huber, Kanade, Sheikh), machine learning (Bagnell), and AI/cognitive psychology (Lebiere).

The vision faculty continued to develop opportunities for computer vision applications in the area of mobile systems, e.g., unmanned ground vehicles, intelligent cars. For example, motion analysis and tracking techniques are instrumental in systems for intelligent driving (Kanade, Sheikh); 3D scene interpretation (Hebert, Huber) is central to the development of unmanned ground vehicles (UGVs). Existing and recent activities include industry-funded projects, and large DoD efforts in the area of UGVs, in collaboration with NREC researchers.

The vision faculty has built on new opportunities in the general area of using computer vision to enhance communication and interaction. For example, owing to progress in face analysis, new modes of interactions combining face analysis and synthesis are being investigated (Cohn, Kanade, Sheikh), at the boundary between computer vision and computer graphics. In the HRI area, research involves the use of sensor fusion and activity recognition to optimize the efficiency of industrial workcells to allow people and intelligent and dexterous machines to work together safely (Huber, Rybski).

The QoLT ERC provided a new set of challenging vision tasks in the areas of recognition and behavior understanding in the general area of assistive technologies. This area provided new opportunities for collaboration with practitioners in rehabilitation science, aging, nursing, and related fields at the University of Pittsburgh and at CMU, giving the vision researchers a better understanding of real-world applications and access to data and users studies. In the last two years, in particular, the impact of this work was evidenced by a range of programs addressing human-centric applications of major impact, e.g., participation in a NSF Expeditions for research on autism (Kanade), computer vision for sensory substitution (Sheikh), and behavior and face analysis (De la Torre).

New exciting opportunities are being pursued in biological engineering in which the expertise gained in the motion analysis and tracking areas led to development of a fully-automated computer vision-based cell tracking algorithms which can track whole populations of cells on a test chip in real-time (Kanade). This approach has the potential of transforming key aspects of biological engineering by reducing the high cost and long timelines for gathering and interpreting experimental data.

All of these activities are supported by strong ties to other units, which we continuously strengthen. Given the importance of machine learning in computer vision, strong ties have been established with faculty in this area both in RI, CSD, and MLD. For example, many of the current vision projects are joint with machine learning faculty (Bagnell, Guestrin, Gordon). In human-centric and medical activities, the involvement of colleagues in psychology, biology, and rehabilitation and related disciplines (University of Pittsburgh) is critical. In mobile systems, projects are collaborative with researchers at NREC (Stentz, Kelly). 3D mapping and inspection projects are based on long-standing collaborations with CEE in (Akinci, Bielak, Garrett). In the area of computer graphics, the Institute had identified the importance of the synergy between computer graphics and computer vision which had led by 2005 to the recruitment of Efros and Narasimhan and more recently of Sheikh. Since then, the graphics/vision has been extremely successful as evidenced by the publication record at SIGGRAPH, which is the venue of record for computer graphics publications, by the volume of media coverage and the students graduated in this area. Finally, collaboration with external organizations through adjunct appointments continued to be an important component of the computer vision work, for example, in the face and behavior analysis (Cohn, Univ. of Pittsburgh, Lucey, CSIRO) and motion and object recognition (Sukthankar, Chen, Intel).

Image segmentation

Image segmentation is a low-level method but usually it is the most important stage in computer vision. Middle-level and high-level algorithms good functionality depends on a part of the quality of image segmentation process results. When an image is acquired from camera sensor the image segmentation process is used as an information reduction or an information enhancing[12]. Image segmentation methods usually have a closely relationship with colour image representation model. Then before take a full overview of segmentation techniques used on robot a full reflection of the colour model chosen in computer vision are going to be taken.

SLAM

A robot must estimate its location relative to its environment, while simultaneously avoiding any dangerous obstacles[13]. This problem is often known as simultaneous localization and mapping (SLAM) or visual odometry.

Statistical techniques used for approximation include Kalman filters, practice filters (aka. Monte Carlo methods) and scan matching of range data. They provide an estimation of the posterior probability function for the pose of the robot and for the parameters of the map. Set membership techniques are mainly based on interval constraint propagation[14]. They provide a set which encloses the pose of the robot and a set approximation of the map. Bundle adjustment is another popular technique for SLAM using image data, which jointly estimates poses and landmark positions, increasing map fidelity, and is used in commercialized SLAM systems such as Google's Project Tango.

New SLAM algorithms remain an active research area, and are often driven by differing requirements and assumptions about the types of maps, sensors and models as detailed below. Many SLAM systems can be viewed as combinations of choices from each of these aspects.

The SLAM process consists of a number of steps.

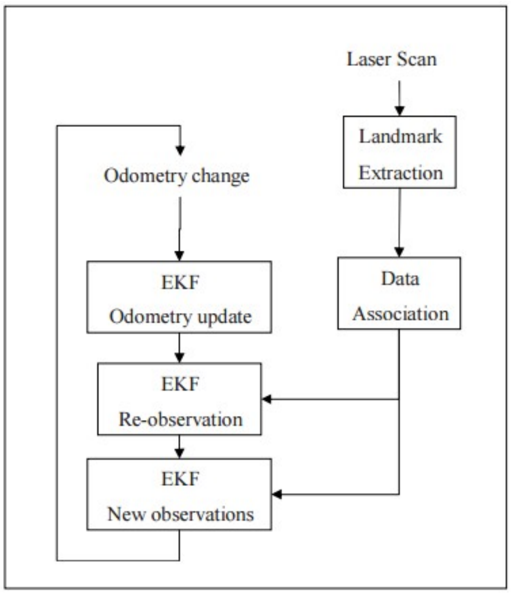

Figure 3.9 Overview of the SLAM process

The goal of the process is to use the environment to update the position of the robot. Since the odometry of the robot (which gives the robots position) is often erroneous we cannot rely directly on the odometry. We can use laser scans of the environment to correct the position of the robot. This is accomplished by extracting features from the environment and reobserving when the robot moves around. An EKF (Extended Kalman Filter) is the heart of the SLAM process. It is responsible for updating where the robot thinks it is based on these features. These features are commonly called landmarks and will be explained along with the EKF in the next couple of chapters. The EKF keeps track of an estimate of the uncertainty in the robots position and also the uncertainty in these landmarks it has seen in the environment. An outline of the SLAM process[15] is given on figure 3.9.

Landmarks

Landmarks are features which can easily be re-observed and distinguished from the environment. These are used by the robot to find out where it is (to localize itself). One way to imagine how this works for the robot is to picture yourself blindfolded. If you move around blindfolded in a house you may reach out and touch objects or hug walls so that you don't get lost. Characteristic things such as that felt by touching a doorframe may help you in establishing an estimate of where you are. Sonars and laser scanners are a robots feeling of touch.

Once we have decided on what landmarks a robot should utilize we need to be able to somehow reliable extract them from the robots sensory inputs.

The type of landmarks a robot uses will often depend on the environment in which the robot is operating.

Landmarks should be re-observable by allowing them for example to be viewed (detected) from different positions and thus from different angles. Landmarks should be unique enough so that they can be easily identified from one time-step to another without mixing them up. In other words if you re-observe two landmarks at a later point in time it should be easy to determine which of the landmarks is which of the landmarks we have previously seen. If two landmarks are very close to each other this may be hard.

Landmarks you decide a robot should recognize should not be so few in the environment that the robot may have to spend extended time without enough visible landmarks as the robot may then get lost.

If you decide on something being a landmark it should be stationary. Using a person as a landmark is as such a bad idea. The reason for this criterion is fairly straightforward. If the landmark is not always in the same place how can the robot know given this landmark in which place it is.

The key points about suitable landmarks are as follows:

- landmarks should be easily re-observable;
- individual landmarks should be distinguishable from each other;
- landmarks should be plentiful in the environment;

- landmarks should be stationary.

Landmark extraction

Once we have decided on what landmarks a robot should utilize we need to be able to somehow reliable extract them from the robots sensory inputs.

There are multiple ways to do landmark extraction and it depends largely on what types of landmarks are attempted extracted as well as what sensors are used.

Spike landmarks

The spike landmark extraction uses extrema to find landmarks. They are identified by finding values in the range of a laser scan where two values differ by more than a certain amount, e.g. 0.5 meters. This will find big changes in the laser scan from e.g. when some of the laser scanner beams reflect from a wall and some of the laser scanner beams do not hit this wall, but are reflected from some things further behind the wall.

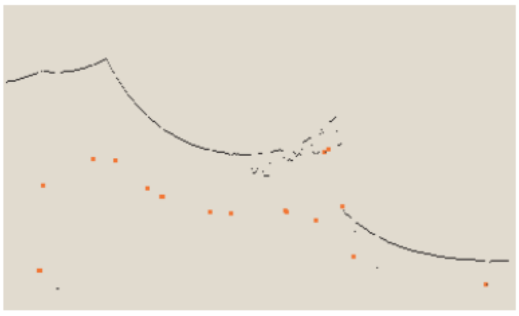
Figure 3.10 Spike landmarks. The red dots are table legs extracted as landmarks.

RANSAC

RANSAC (Random Sampling Consensus) is a method which can be used to extract lines from a laser scan. These lines can in turn be used as landmarks. In indoor environments straight lines are often observed by laser scans as these are characteristic of straight walls which usually are common.

RANSAC finds these line landmarks by randomly taking a sample of the laser readings and then using a least squares approximation to find the best fit line that runs through these readings. Once this is done RANSAC checks how many laser readings lie close to this best fit line. If the number is above some threshold we can safely assume that we have seen a line (and thus seen a wall segment). This threshold is called the consensus.

The below algorithm outlines the line landmark extraction process for a laser scanner with a 180° field of view and one range measurement per degree. The algorithm assumes that the laser data readings are converted to a Cartesian coordinate system – see Appendix A. Initially all laser readings are assumed to be unassociated to any lines. In the algorithm we only sample laser data readings from unassociated readings.

Camera calibration

Camera projects 3D world onto the 2D image plane. Finding the quantities internal to the camera that affect this imaging process: image center, focal length, lens distortion parameters[16]. The use of a calibration pattern or set of markers is one of the more reliable ways to estimate a camera's intrinsic parameters. In photogrammetry, it is common to set up a camera in a large field looking at distant calibration targets whose exact location has been precomputed using surveying equipment. In this case, the translational component of the pose becomes irrelevant and only the camera rotation and intrinsic parameters need to be recovered. If a smaller calibration rig needs to be used, e.g., for indoor robotics applications or for mobile robots that carry their own calibration[17] target, it is best if the calibration object can span as much of the workspace as possible (Figure 3.1), as planar targets often fail to accurately predict the components of the pose that lie far away from the plane. A good way to determine if the calibration has been successfully performed is to estimate the covariance in the parameters and then project 3D points from various points in the workspace into the image in order to estimate their 2D positional uncertainty.

The traditional way to calibrate a camera is to use a 3D reference object[18]. In Fig. 3.11, the calibration apparatus used at INRIA[19] is shown, which consists of two orthogonal planes, on each a checker pattern is printed. A 3D coordinate system is attached to this apparatus, and the coordinates of the checker corners are known very accurately in this coordinate system. A similar calibration apparatus is a cube with a checker patterns painted in each face, so in general three faces will be visible to the camera. Figure 3.12 illustrates the device used in Tsai's technique[20], which only uses one plane with checker pattern, but the plane needs to be displaced at least once with known motion. This is equivalent to knowing the 3D coordinates of the checker corners. A popular technique in this category consists of four steps[19]:

1. Detect the corners of the checker pattern in each image;
2. Estimate the camera projection matrix P using linear least squares;
3. Recover intrinsic and extrinsic parameters A, R and t from P;
4. Refine A, R and t through a nonlinear optimization.

Note that it is also possible to first refine P through a nonlinear optimization, and then determine A, R and t from the refined P. It is worth noting that using corners is not the only possibility. We can avoid corner detection by working directly in the image. In [21], calibration is realized by maximizing the gradients around a set of control points that define the calibration object. Figure 1.4 illustrates the control points used in that work.

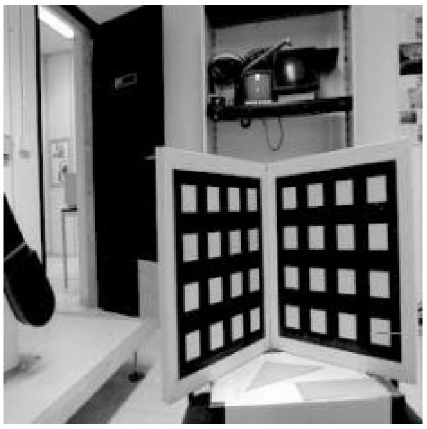
Figure 3.11

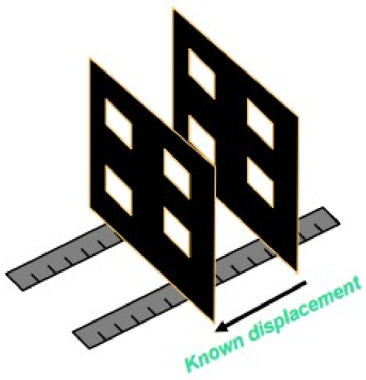

Figure 3.12

4 Overview of task specification and project schedule

The table below is a project schedule. It contains tasks which were or must be completed. If a primary task is quite complex, it is partitioned into the sub-tasks. Start date column shows the date, when a task was started or when it is supposed to be started. Duration column shows approximate time (in hours) that was or is needed to complete the task. And the final column shows the difficulty (experienced or expected) of the task for the author of this paper from 1 (very easy) to 10 (extremely hard).

Table 4.1 Project schedule

№	Task	Sub-tasks	Start date	Duration (hours)	Difficulty (1-easy – 10-hard)
First part					
1	Install ROS[22]	a) Install Ubuntu	10.09.2016	3	1
		b) Install ROS	10.09.2016	2	2
2	Learn ROS	-	10.09.2016	20	3
3	Learn simulator (Gazebo)	-	19.09.2016	15	3
4	Learn how to connect to physical DARwIn-OP2	a) ssh wired	23.09.2016	1	1
		b) ssh wireless	30.09.2016	2	2
5	Learn how do python scripts work for manipulating DARwIn-OP2	-	15.10.2016	30	5
6	Write your own script that makes DARwIn-OP2 push something	-	03.11.2016	5	6
7	Create Android App for manipulating DARwIn-OP2 via Wi-Fi	a) Create GUI	03.12.2016	10	3
		b) Create architecture of an app	04.12.2016	20	5
		c) Establish connection between Android device and DARwIn-OP2 to control robot	10.12.2016	10	4
8	Create DARwIn-OP2 architecture in Python for convenient manipulation	a) Architecture	13.12.2016	30	8
		b) All left actions (attacks, moves)	20.12.2016	40	6
Second part					
9	Teach DARwIn-OP2 to fight with another one	Learn Basics of Computer Vision	03.01.2017	120	9

		Learn to work with DARwIn-OP2's gyroscope	05.02.2017	90	7
		Create algorithm for finding and attacking an aim	05.03.2017	160	8
		Teach DARwIn-OP2 to defend from attacks of another DARwIn-OP2	06.04.2017	200	9

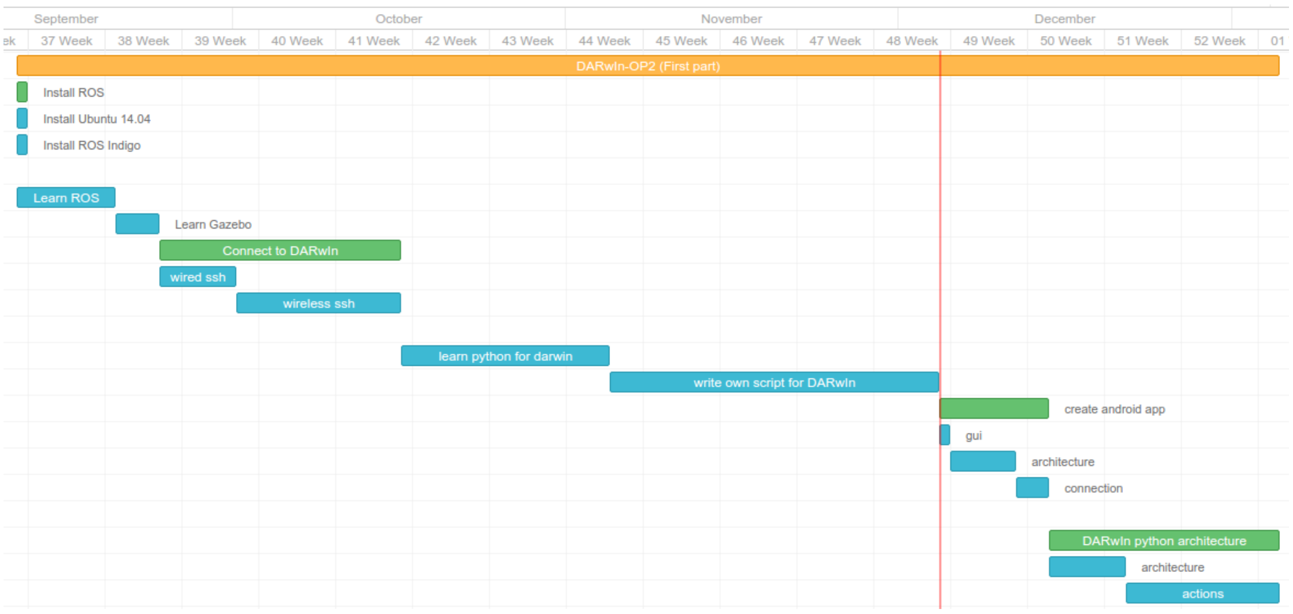

Figure 4.1 Gantt's graph for the first part

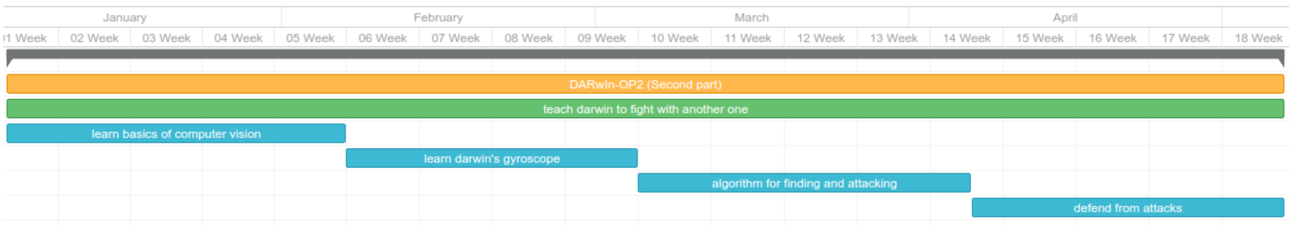

Figure 4.2 Gantt's graph for second part

5 Review of tasks

Task 1: done. There were no problems.

Task 2: on-going. For now I've learned everything what I need to write scripts for DARwIn on python and to manipulate it using rospy library.

Task 3: started, not finished, and I'm not going to finish it, because my computer is too old for such software, and there is a real DARwIn available for testing.

Task 4: done. The most easiest task. Wireless/wired connection with DARwIn can be established by writing no more than ten commands in Ubuntu terminal.

Task 5: in progress. I know how to perform simple actions like push, stay, sit, and so on. Now I'm learning how the walking algorithm works and DARwIn is balancing.

Task 6: done. One of the easiest algorithms. I had a problems when I was setting the wrong angles and motors were blocking when they met obstacles, so that I had to reset DARwIn.

Task 7: not started. It won't be very hard task to create Android application for manipulating DARwIn. The only problem which will take a lot of time is thinking about good extensible architecture.

Task 8: not started. Again the only weighty task is to create a good architecture, such that I will have less probability to rewrite it later. And another sub-task which will take a lot of time is writing scripts for actions. Now I see it as a tedious routine. But I will spend a lot of time thinking about good architecture of project, so that they won't be routine.

Task 9: not started. The most hard task for me. A lot of areas of robotics that I don't know. I think that I'll even have to implement some of genetic algorithm and spend a lot of time for it.

6 Interim results

Firstly, the ROS Indigo was installed. Since Ubuntu of version 14.04 is required for that, it was also installed.

I spend some time to learn ROS: navigating the ROS file system, creating ROS packages, building ROS packages, understanding ROS nodes, understanding ROS topics, understanding ROS services and parameters, using rqt_console and roslaunch, using rosed to edit files in ROS, creating ROS msg and srv, writing ROS publisher and subscriber (python) and so on.

I learned a little about Gazebo simulator which can simulate DARwIn-OP2. But I didn't learn it well because didn't have a computer with requirement parameters, and we have a real one in the lab.

After that, to teach DARwIn do something I needed to establish connection with it. Connect using ssh was easy. DARwIn's address was 162.198.123.1. Bot for wireless connection I had to download an extra software for ubuntu – vncviewer. It allows to open a desktop of remote computer on a computer, which vnciviewer runs on. In vncviewer I connected DARwIn to the same wireless network, which my computer was connected to, and after that, using ifconfig linux terminal command, I needed to find out what was the inet address of wlan0 in DARwIn. After that, using this address, I've got an ability to connect to DARwIn using wireless network via ssh.

When I've connected to DARwIn, I could see, how the python scripts that use rospy library work. Exactly on the example of walker script.

Using that script as an example, I've written my own one, that makes DARwIn push something. It is not perfect, but I'm going to handle it.

On the photos below one can see DARwIn-OP2 performing some actions (it is the result of connecting to DARwIn-OP2 using wired/wireless ssh connection, creating workspace in it with python scripts that make DARwIn perform these actions):

Image 6.1: initial position of DARwIn after switching on.

Image 6.2: DARwIn is standing up.

Image 6.3: DARwIn pushes imaginary rival.

Image 6.4: DARwIn in battle stance.

Image 6.1

Image 6.2

Image 6.3

Image 6.4

7 Short term plans

In my future plans are included: learning Android, computer vision, python (especially rospy library), spending a lot of time in the laboratory, read a lot of papers related to robotics and finish this project with excellence.

At the end of my work DARwIn-OP2 will be able to fight with another one: throw punches, legkicks, defend from attacks. Also it can be controlled by a specific application on Android device, which will also be implemented.

References

[1] Wikipedia, the free encyclopedia (DARwIn-OP): https://en.wikipedia.org/wiki/DARwIn-OP.

[2] "Me And My Robot Page 2 of 2". Forbes.com. 2011-05-23. Retrieved 2011-12-27.

[3] Author Name: Dennis Hong (2011-05-12). "Robotis DARwIn-OP Raises The Bar | Robot Magazine - The latest hobby, science and consumer robotics, artificial intelligence". Find.botmag.com. Retrieved 2011-12-27.

[4] Darwin OP2 specifications and comparison with DARwIn OP.

[5] Development of Open Humanoid Platform DARwIn-OP.

[6] Darwin-OP2 Robot To Help Children With Autism Communicate More Easily.

[7] ICRA 2015 DARwIn-OP Humanoid Application Challenge – Indiana Darwin.

[8] Efficient and Versatile Modular Configuration.

[9] Walking Control Algorithm of Biped Humanoid Robot on Uneven and Inclined Floor (Jung-Yup Kim, Ill-Woo Park and Jun-Ho Oh).

[10] J. Y. Kim, I. W. Park, J. H. Oh.: Experimental realization of dynamic walking of the biped humanoid robot KHR-2 using zero moment point feedback and inertial measurement. Advanced Robotics. 20(6): 707-736 (2006).

[11] Spatiotemporal Bundle Adjustment for Dynamic 3D Reconstruction.

[12] Artificial Visionin the Nao Humanoid Robot.

[13] Computer Vision: Algorithms and Applications; Richard Szeliski; September 3, 2010; draft 2010 Springer.

[14] A Nonlinear Set Membership Approach for the Localization and Map Building of Underwater Robots (Luc Jaulin).

[15] SLAM for dummies. A Tutorial Approach to Simultaneous Localization and Mapping.

[16] Robotics 2. Camera Calibration. Barbara Frank, Cyrill Stachniss, Giorgio Grisetti, Kai Arras, Wolfram Burgard (Uni Freiburg).

[17] IEEE transactions on robotics and automation, vol. 7, №.1, February 1991 (Brian J. Waibel and H. Kazerooni, Member, IEEE).

[18] Camera calibration

[19] Olivier Faugeras. Three-Dimensional Computer Vision: a Geometric Viewpoint. MIT Press, 1993.

[20] Roger Y. Tsai. A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf tv cameras and lenses. IEEE Journal of Robotics and Automation, 3(4):323–344, August 1987.

[21] L Robert. Camera calibration without feature extraction. Computer Vision, Graphics, and Image Processing, 63(2):314–325, March 1995. also INRIA Technical Report 2204.

[22] ROS Indigo