Exploring Behaviors & Collaborative Mapping through Mindstorms Robots: A case study in applied social constructionism at senior-project level

Nikolaos Mavridis, Asma Al Rashdi, Maryam Al Ketbi, Sara Al Ketbi, Alia Marar
Interactive Robots and Media Lab, United Arab Emirates University, Al Ain, UAE
NikolaosM@uaeu.ac.ae

Abstract

Constructionism, a term first coined by Seymour Papert, is a learning theory based on constructivism, which however also holds that learning can happen most effectively when people are also active in making objects in the real world. In this paper, we will present a case study of an application of constructionism at the undergraduate senior project level. More specifically, we will describe a robotics project that took place in our lab, which aimed towards providing a strong hands-on background in the basics of robotics and the fundamentals of the research process to a group of four final-year students. During this project, the students experienced basics of team working, flexible project management, and intra- as well as extra-group constructionist tuition, as well as aspects of real-world research. Furthermore, they were able to gain experience in three programming languages, build and successfully demonstrate basic behaviors and collaborative mapping using the Mindstorms robots, and create a theoretical framework incorporating and providing novel extensions to their methods.

1. Introduction

The creation of *effective undergraduate computer science and engineering* curricula and methodologies, is quite a difficult problem, given a number of issues. First, there is a clear need for a sound theoretical base, on which advanced-year subjects can build upon; second, the nature of the fields as well as the requirements of the market is quite broad as well as constantly changing; and third, quite importantly, apart from the purely technical skills, we would like our graduates to have adequate real-world basic experiences of team working, project management, research, as well as constructionist facilitation.

Within almost all undergraduate curricula, the *senior project course* is of central importance. This normally takes place at the final year of studies; and provides the last opportunity for the students to experience integration, application, and extensions of their previously acquired skills towards a well-defined project of considerable length (usually 1-2 semesters).

When designing or advising such a course, a *number of important choices* need to be made. The first is concerned with number of students per project; should there be a single student working towards each project, or maybe a group of two or four? Yet another choice, is concerned with the basic educational methodology applied – which is usually closely guided by the learning theory that one chooses to adopt.

Learning theories have existed since ancient times. The probably oldest theory is usually referred to as the "Transmission Model". However, adopting this theory usually leads to the traditional models of teaching: uni-directional lectures providing ready-made knowledge to passive learners who focus on rote memorization. In contrast, most modern theories, lead to active learners, as we shall discuss further.

Having mentioned goals, duration, group size, and learning theory, one needs to also make decisions regarding the *actual content* of the senior project course, and the *platforms or tools* utilized as part of it.

After this brief introduction to the overall setting of the problem, we are ready for the specifics: in this paper, we will provide a *case study* of the application of the learning theory of social constructionism, towards achieving a number of educational goals for the case of a final-year senior project focusing on robotics, carried out by a group of four students, and utilizing Mindstorms robots.

We will start, in section 2, by providing background on constructionism, mindstorms robots and education, and on basic behaviors, odometry, and mapping. In sections 3-6, we will describe the work carried out during the project. Finally, in section 7 we will provide an extensive discussion and close with forward-looking concluding comments and suggestions.

978-1-4244-5699-4/10/$26.00 ©2009 IEEE

2. Background

Social Constructionism and Learning Theories: Learning theories have existed since ancient times, albeit not necessarily in an explicit and formal manner. Historically, learning theories start with the so-called "Transmission Model" (for characteristics see [1]), according to which, knowledge is transmitted from the mind of the teacher, through words, to be imprinted to the mind of the student. However, adopting this theory usually leads to the traditional models of teaching: uni-directional lectures providing ready-made knowledge to passive learners who focus on rote memorization. In contrast, most modern theories, view knowledge as not being something that is to be transmitted; but rather, which is constructed within the learner's mind – and thus, lead to active learners.

Exactly this notion, namely that knowledge is constructed within the learner's mind on the basis of his existing knowledge as well as his new experiences, is central to the theory of "Constructivism", first formalized and detailed by Jean Piaget (see [2]). The implications for pedagogy are immense; the teacher is no more a transmitter of knowledge; but rather, a facilitator, enabling educational experiences for knowledge construction within the student's mind by appropriate scaffolding. An extension of constructivism, once the focus moves from the individual to the group, is "Social Constructivism". According to this theory, it is within the group, and not just the individual mind, where knowledge is constructed. Pedagogically, this stance usually also leads to the creation of small groups, and the facilitation of appropriate activities for knowledge construction. Furthermore, within-group discussion, demonstration and co-facilitation is highly encouraged.

Although the main mode of knowledge generation (reception vs. construction) as we move from the transmission to constructivism, and the external environment starts to enter the picture by providing experiences other than just the teacher's voice, we are still missing one more ingredient in order to introduce "Constructionism" [3]. We are missing an object to be built; constructionism adopts constructivist views, but also holds that learning can happen most effectively when people are also active in making objects in the real world. Thus, the social version of constructionism is the main learning theory adopted in our project.

Mindstorms Robots and their Educational Use: Mindstorms [4] a line of Lego sets containing programmable bricks as well as electric motors, sensors, Lego bricks, and Lego Technic pieces (such as gears, axles, and beams), out of which many different kinds of robots can be built. Mindstorms is named after the book Mindstorms: Children, Computers, and Powerful Ideas [5] by the MIT Media Lab faculty S. Papert, the creator of the Logo language and the constructionist learning theory.

Mindstorms have an established history of educational use; a characteristic set of examples can be found in [6]-[9]. As can be seen, in the past they have been utilized towards a variety of target areas: AI, computer engineering, control theory, mechatronics, CS, all the way to precision agriculture. Furthermore, various educational levels were targeted, from high-school to graduate, various educational paradigms were used: constructionism, peer learning, problem-based learning, and various locations of course delivery have been reported: US, Japan, Europe etc. However, no reports of their use towards collaborative mapping exist in the literature, and certainly not within senior-project context under a social constructionist perspective in a gulf country.

Basic explored issues of Mobile Robotics:

In our project, the students dealt with a number of issues regarding robotics, including body design, programming language choice (graphical [10], C-like [11], special Java [12]), basic behaviors of {approach, avoid, keep-distance, explore}, as well as human-robot-wall recognition, odometry, and mapping.

3. Overview of project

Now, having briefly discussed relevant background, we will provide an overview of the project and the next sections. The total duration of the senior project course at our department is roughly 16 weeks. The student group of 4 started familiarizing themselves with NXT one month before the official course start. Then, under minimal facilitation, they decided to create an overall timeplan, as well as to assign roles and responsibilities in the group. They broke down the project to three phases (A-C) of roughly one month each, as we shall see below. From the beginning, it was clear to them that there might be overlap between the phases. However, realizing that they are slowly moving on from the rigidity of pre-planned lab-type exercises towards the fluidity of real-world longer term projects, they decided that this was not necessarily discomforting, irrespective of the initial uneasiness. Also, a schedule of hours with meeting with their advisor was devised, again flexible, but generally on the order of at least 4-6 hours weekly. It was decided that the team would also meet alone usually in a room near the lab, for ease of seeking help from their advisor if required. In more detail:

4. Phase A: Requirements and Analysis

The students began investigating the NXT robot and the specs of possible add-ons. These specifications helped to frame the work under the available features:

a) Sensors: NXT comes with four basic sensors, and there are also many others at the market: for example, the Vision Subsystem [13]. We researched these sensors to explore possible functions.

b) Motor multiplexing: This study was done in order to guarantee the possibility of expanding the number of the motors [14], above the standard 3.

c) Programming Languages: MS Robotics St, Lejos, iCommand, RobotC, C# were tried out. At our project we used NXT-G, robotC, and iCommand to accomplish our goals. MSRS suffered from lack of documentation, RobotC from inherent limitations.

d) Physical model design: Researched, and decided to use the vehicle and tribot types, as fit to our goals.

e) Charting API: to be used for mapping - jFree Chart was a good solution [15]

5. Phase B: Architecture and Design

In this phase, body designs, as well algorithms, were architected, in order to be able to support the requirements set, for the two main functional goals: basic behaviors and collaborative mapping. It is worth noticing that the tight interplay of the physical (body) with the virtual (code) became apparent to the students, as did the concept of hardware/software co-design.

5.1. Basic Behaviors Body & Algorithms

By basic behaviors here we refer to triggerable behaviors of: approach, avoid, keep distance, and explore. The basic behaviors are triggered on the basis of stimulus type (wall, ball, human, robot), which are recognized on the basis of their ultrasound profile.

a) Robot Physical Design for Basic Behaviors:

The robot design for the behaviors is a car-like model. (figure 1L), built with a pair of fixed driving wheels, a front pair of steering wheels, and a front ultrasonic. Rationale: Allows rotating the sonar to scan the object width, simplicity, stability, minimizes the position errors [16].

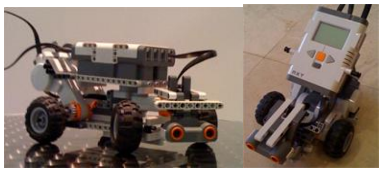

Figure 1: Car-like (L) and TriBot (R) Bodies

b) Algorithms for Basic Behaviors:

Object recognition: The robot recognizes the type of the object (ball, human, robot, or wall): First, the robot head (sonar) is rotated, while facing the object. By thresholding the sequence of distances, the angular width of the object is measured. Then, based on an investigation we carried out in order to estimate the probability that an object with a given angular width at a given distance belongs to each of the categories (ball vs robot etc), we recognize the object.

Keep Distance behavior: the robot goal is to keep distance within a range. If the robot is moving and the object is sensed within the specified range, the robot will stop. If the object gets close to the robot, the robot will move backward, if the object moves away the robot will move forward to the object.

Avoid behavior: The robot is initially moving, and stops while sensing an object within the specified range. It avoids the object after finding the best direction that separates the robot from the object.

Approach and chasing behavior: The goal is to get closer to an object. The robot moves toward and stops while sensing an object within specific range. It moves toward after doing some calculation to find the appropriate angle that makes the robot approach.

5.2. Mapping Body & Algorithms

After behaviors, our second goal was to experiment on collaborative mapping of the environment:

a) Robot Physical Design: In the Mapping stage, robot creates a map for the surrounding area. The robot design is changed to TriBot, figure 1 (R). The driving system for this design called differential drive [16]. It is based on a two wheels drive system with autonomous motors to turn each wheel, and one non-driven wheel. This design supports a wider range for scanning, which is not provided by the car-like design.

b) Algorithms for Mapping:

The robot will move within grid points. Basic steps:

1) PC sends request to the robot (Bluetooth) to move

2) The robot scans the surrounding area of each position it reaches in a circular path.

3) In each position while scanning, PC calculates the robot position and scanned points.

4) The robot moves from one position to another randomly based on the chosen angle.

5) Map is built based on motor enc. and sonar data.

The host PC sends a request to the robot to move, which turns with specific angles, and scans the area using the sonar to find out the existed object. The calculated measurements by the host PC are used to present the map graphically.

6. Phase C: Implementation and Testing

6.1. Basic Behaviors Implementation & Tests

a) Physical Model Construction: We utilized:
- Three servo motors and four tires: The degree of freedom of the front wheels is 40 degree in each side.
- One Ultrasonic Sensor, and a set of beams, pins, axles, cables and other Mindstorms NXT accessories.

b) Software Implementation: RobotC code parts:
Pattern Rec: The robot measures the angular width of the detected objects by rotating the sonar to left and right. Then, based on the total angular width it determines the object's type. For example, if the ang. width >30 and <55, the detected object could be a ball.
Keep Distance: This behavior is implemented based on a continuous reading from Ultrasonic. It maintains the target in a range (25-35 cm) with fixed theta (0).
Obstacles Avoidance: Basic steps are:
S1 Estimation stage: in this stage the robot should determine the required distance to avoid the detected obstacle and the best angle to rotate as well.
S2 Selection stage: In this stage we select the angle to avoid the obstacle based on the minimum distance.
S3 Action stage: the robot will turn the selected angle from the previous stage. Then, it will move backward for specific distance. After moving backward, the robot will do some turns to move to another direction.
Target Chasing: The steps are:
S1 Estimation stage: in order to detect the location of the moving target the robot will scan the front surrounding scene of 60 deg in both L/R sides. While rotating, the robot will record what it has detected.
S2 Selection stage: the robot selects the angle with the minimum distance (where object is existed).
S3 Action stage: the robot will rotate the selected theta and move for specific distance.

c) Testing: We run each of these behaviors separately, and then, observed the activation of various behaviors based on stimulus. The robot sometimes lost moving targets due to limited sonar range, but performed well in object recognition.

6.2. Mapping Implementation & Tests

a) Physical Model Construction: Parts:
- Three servo motors, two tires, and one castor: the castor is a non-driven wheel that supports the body.
- One Ultrasonic Sensor, and set of beams, pins, axles, cables & other Mindstorms NXT accessories.
The robot has the following geometry which was used to control its motion: tires circumference 17.584cm, distance between tires 11 cm, body circumf. 34.54 cm.

b) Software Implementation: This phase parts are implemented using iCommand Java library:
Connection & robot-computer Interaction: we have used iCommand methods to establish the connection over Bluetooth. The interaction is done in the two directions: from PC to the robot and vice-versa.
Robot motion control: Created methods for relative as well as absolute robot movements:
Robot odometry: this technique uses both motors' encoder values to determine the robot positions relatively from the starting point (x, y, θ, t,), where t is the timestamp of the position, with an update rule like:

$$x[t+1] = x[t] + d\cos(\theta),\ y[t+1] = y[t] + d\cos(\theta)$$

The same approach is used to measure the positions (x, y, θ, t) of objects in the environment. We maintain a record of robot as well as object positions for every t.
Occupancy Map Construction: the grid map infrastructure is constructed using a two dimensional array. Each cell indicates whether the corresponding to one block of 1cm2 in the real work occupied or empty. The map will be updated based on the robot's path and detected objects; hence, no prior knowledge about the environment is assumed. Besides, it is manipulated using Bresenham's line algorithm[17]. The cells values are given a certainty value to indicate whether the robot or the object is likely to exist there or not. Initial cell value is 0.5 which indicates that the area hasn't been explored yet. The updating algorithm will gradually change the value between 0…1. Real time map is drawn while the robot is navigating.
Collaborative mapping: i.e., the aggregation of two robots' occupancy maps. Each robot will explore a region and will keep a record of its position and the environment's objects. At the beginning, each robot will rotate 360 degree while scanning the surrounding after each 30 degree in order to check if any object is located nearby. Then, the robot will randomly choose one of four directions and it must be free: 0, 90, 180, or 270 to pursue to a new grid point. The records of the two robots are merged to create a global map.

c) Testing: Odometry and Collaborative mapping tests were carried out.

d) Theoretical Framework for extensions:
We devised a 3-level framework of possible goals towards extensions (empirical and math analysis in future paper), by reflecting on collaborative mapping:

L1) Cover an area: Cover whole area/create map asap
L2) Keep updated info: We assume that mapping loses confidence over time, and decide where to revisit
L3) Revisit often where changes occur: Keep an estimate of observed variance, and decide where to go

7. Discussion and Conclusion

During this project, through appropriate facilitation, the students acquired experiences as well as skills, from quite a wide range: all the way from the technical, to team working and project management.

In terms of the *technical*, basics of robotics such as sensing and actuation, as well as robot body design, interplay with software, basic behaviors, odometry, and multi-robot mapping and communication were covered with a creative hands-on approach. Experience with multiple languages, as well as goal-based cross-comparison and selection, was acquired.

In terms of other *not-so-technical* skills and experiences: first, aspects of team working, such as co-operation, co-facilitation, role and responsibility assignment and negotiation, as well as aspects of leadership were illuminated. Second, flexible project management was experienced: a fluid construct of three overlapping phases at the macro level, along with multiple maneuvers and replanning at the micro level, was devised and utilized. Third, at the research skills level, the students were not only able to lay out basic steps and go through cycles of incremental design, implementation, and testing, but they experienced simple literature search as well as the creation of a theoretical framework unifying and providing a pathway for extensions to their work.

Finally, clearly the case study described in this paper, provides us with a *real-world example* of successful application of social constructionism: a team of four students, facilitated but not directed, with an external object to build upon (physically, the robots kit, and virtually, the programming environments and the goals set), were able to self-organize and go through a multi-faceted educational experience, which catalyzed the construction of very valuable new technical and not-so-technical knowledge, ranging from robotics and software to team working and project management. In that respect, we think it provides for an interesting case study, which can inform future projects in many ways: not only in terms of the specifics of the project design (duration, team size, phases, hardware and software), but also, in terms of the concepts that can be illuminated.

In conclusion, this project was a life-changing experience for the students, and an important prototype for future projects for the advisor. We hope that it can also act as a prototype for other similar projects around the world, acting as an exemplar for the benefits of adopting an applied social constructionist stance.

References

[1] G. Claxton, "Teaching to Learn: a direction for education", London: Cassell, 1990

[2] C. T. Fosnot, Ed. "Constructivism. Theory, Perspectives, and Practice", New York: Teacher's College Press, 1996

[3] E. Ackerman, "Piaget's Constructivism, Papert's Constructionism: What's the difference?", retrieved Nov.09: http://learning.media.mit.edu/content/.../EA.Piaget%20_%20Papert.pdf

[4] LEGO Mindstorms website, see wikipedia and also www, retrieved Nov.09 from: http://mindstorms.lego.com

[5] S. Papert, "Mindstorms. Children, Computers and Powerful Ideas", New York: Basic books, 1980

[6] S. Parsons, E. Sklar, "Teaching AI using LEGO Mindstorms", in AAAI Spring Sympos. 2004 on Accessible Hands-on Artificial Intelligence and Robotics Education

[7] J.B. Schafer, "Hands-on Artificial Intelligence Education Using LEGO Mindstorms: Lessons Learned", in MICS 2004

[8] M. Mota, "Using Lego Mindstorms and Robolab as A Mean To Lowering Dropout and and Failure Rate In Programming Course", in Frontiers in education conf. 2007

[9] J. Bergin, R. Lister, at al., "The First Programming Course: Ideas to End the Enrollment Decline", in SIGCSE06

[10] NXT-G graphical blocks language tutorial, see www: http://www.ortop.org/NXT_Tutorial/ (retrieved Nov.09)

[11] RobotC language, see http://www.robotc.net

[12] LeJOS Java for Mindstorms, http://lejos.sourceforge.net

[13] NXT-Cam Vision Subsystem. Retrieved from www: http://www.mindsensors.com/index.php?module=pagemaster&PAGE_user_op=view_page&PAGE_id=78 (Nov.09)

[14] MTRMX-NX MotorMux. Retrieved Nov09: http://www.mindsensors.com/index.php?module=documents&JAS_DocumentManager_op=viewDocument&JAS_Document_id=9

[15] jFreeChart free charting API. Retrieved from www: http://www.jfree.org/jfreechart/ (retrieved Nov.09)

[16] Rao, "What type of robot to choose? P II". From www: http://www.roboticsindia.com/modules.php?name=News&file=article&sid=29 (retrieved Nov.09)

[17] Bresenham's line algorithm. (n.d.). Retrieved April 08: http://en.wikipedia.org/wiki/Bresenham's_line_algorithm