STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **LV**, Number 1, 2010

AN AUTONOMOUS APPROACH TO WHEEL CHANGING PROBLEM

LIVIU ŞTIRB, ZSUZSANNA MARIAN, AND MIHAI OLTEAN

Abstract. We describe a self-repairing robotic car capable of changing its wheels automatically. The robot is constructed from a Lego NXT kit and a Lynx Arm kit which are coordinated from a PC. The difficulty consists in assembling the wheel on its shaft with a high precision which is not possible with the Lego components. This was solved by creating an ensemble made from both wheel and its shaft. The entire ensemble is replaced instead of the wheel alone. We have performed several experiments which show the effectiveness of our approach.

1. Introduction

Changing the broken wheel of a car is an operation that most people try to avoid. When this happens the drivers prefer to call a service company to perform this task. They do that because changing the wheel can generate several problems, some of them being listed below:

- it is time-consuming,
- requires some tools,
- requires some knowledge on how to do it,
- it can make both the hands and clothes very dirty and badly smelling,
- in some cases it can generate injuries to fingers.

For solving these problems we have designed a robotic car which is able to change its wheels autonomously with no interference from humans. The robot consists of 2 main parts: the car itself and an arm placed above the car, which will change the broken wheel. The car is constructed from a Lego NXT kit, while the arm is a standard Lynx arm with 6 degrees of freedom. The difficulty of the problem is due to the low power and low precision of the involved components: it is almost impossible to insert an axel into a motor. Using the Lego and Lynx kits one simply cannot do that.

2010 *Mathematics Subject Classification.* 68T40, 93C85, 70B15.

1998 *CR Categories and Descriptors.* I.2.9 [**Computing Methodologies**]: Artificial Intelligence: Robotics.

Key words and phrases. robot, car, wheel changing, self-repairing, Lego NXT, Lynx arm.

In order to solve this problem we propose a more complex device. Instead of changing the wheel, we change an ensemble composed from wheel and its axel. We also design a mechanism where the wheel and its axel can be inserted without needing a high precision.

The paper is structured as follows: Section 2 surveys some material in the field of self-repairing and self-replicating robots. Section 3 deeply describes the proposed approach. It describes the hardware, the software and all the steps required for changing the wheel. Section 4 discusses the numerical experiments and their outcome. In section 5 we enumerate some of the strong and weak points of our idea. Finally section 6 concludes our paper.

2. Related Work

While there are a lot of articles describing self-replicating, self-reconfiguring, self-constructing and self-assembling robots, there are only a few that present self-repairing robots. One of the best known is presented in [6] and consists of a robotic chair that is capable of falling apart and then reassembling itself again. Building the robot required two years of work. It is made of 14 motors, two gearboxes and many other pieces, it has its *brain* in the seat, and uses a sophisticated algorithm to find its pieces (the four legs and the back of the chair) and eventually re-assembles itself and stands up. The chair, which has first been presented at the IdeaCity conference in Toronto, won the acclaim of the cyberart sophisticates at the 2006 ARS Electronica conference in Linz, Austria [7].

In 2009 researchers from the Pennsylvania University built a modular robot that is capable of self-assembling, after it has been kicked apart [9]. This robot is made of 5 clusters, each cluster being made of 4 modules and a camera, and is capable of moving by itself. Each module has 4 circuit boards and a motor. Some of the modules are connected to the others through screws, while others through magnets. When the robot is kicked, parts connected through magnets separate from each other. Each cluster contains a LED, which displays a custom blinking pattern, and a camera. These two help the clusters to locate and identify each other. Clusters can turn and crawl which helps them to find each other and connect through the magnets. The weak point of this approach is given by the fact that the robot is composed from identical modules which are less likely for real-world approaches.

In [4] a set of modular robot cubes that are capable of self-replication (the capability of constructing a detached, functional copy of itself, which will also be capable of replication) is presented. This robot is made of 10-cm module cubes which have electromagnets that selectively weaken and strengthen the connections, determining where the structure breaks or joins [4]. These cubes are split into halves along the (1,1,1) plan and one half can swivel relative to the other in 120 degrees increments. These cubes are powered through the baseplate and transfer power and data through their faces. This can be considered a disadvantage, because it means that the modules are not functional outside the laboratory. Taking cubes from specific locations a four-module robot was capable to replicate itself in 2.5 minutes, while a three-module robot needed only about a minute [4]. Although this robot is capable of self-replication not self-repair, such systems can be considered self-repairing, because they might be capable of changing a module that is not working properly with a good one. The most important aspect of this approach is the scalability: the replicated robot can contain any number of cubes.

Another interesting self-repairing robot is described in [5]: this robot uses an autonomous and continuous process of self-modeling, thus observing changes in its own morphology and synthesizing new behavior. This process is composed of three algorithmic components, executed continuously by the robot while moving or resting: modeling, testing and prediction. First an arbitrary motor action is performed by the robot and its result is recorded. Using this sensory-actuation casual relationship the model synthesis component creates a set of 15 candidate self-models, and determines the next action, that would give the most information about the robot. After 16 cycles of this, the most accurate model is used to generate the new, desired behavior. The authors present how this approach was tested using a robot with four legs and eight motorized joints, eight joint angle sensors, and two tilt sensors. This robot was allowed to move for a while, after which it suffered some damage (usually a part of one of its legs was removed). Starting from the best self-model from before the accident, the robot was capable of developing a new behavior and it started moving again.

3. Proposed approach

In this section we deeply describe the proposed approach. We start by presenting the involved hardware. Then, we describe the proposed software. Finally, we put all together and we give details on how the robot works.

3.1. **Hardware.** The car is constructed with pieces from the Lego Mindstorm NXT kit. The mechanism that changes the wheel is a Lynx arm with 6-degrees of freedom.

Three servo motors are attached to the Lego brick. Their purpose is:

- The first one is at the back of the robot and is used to move it, using two bigger wheels placed at the back side of the robot. In the current stage of the project these wheels cannot be changed, but we are working on new version where these changes are possible.
- The second motor is used to lift up the robot a little, when the wheel is changed (just like one lifts the car when he changes the wheel). Some

Lego pieces are attached to it, which in normal state (when the robot is moving) do not touch the ground. To lift the car the motor is rotated with 80 degrees and these attached pieces get to touch the ground, moreover lift the front part of the robot a little, so the front wheels do not (or only slightly) touch the ground (see Figure 1).

- The third motor is located at the left side of the robot and is used to move a Lego piece called cross axel, which opens or closes the part that holds the wheel (see Figures 2). If it is open, the left wheel structure can be taken and replaced by another similar one. In order to be able to change both the left and the right wheels at the front of the car, we would need another motor, to work in the same way at the right part of the robot, but that would imply another brick, since the maximum number of motors that can be used with a brick is 3.

The robotic car has a Lynx 6 robotic arm [10] on its top, which is used to change the wheel. This arm has a base, shoulder, elbow, wrist, and a functional gripper. These are important, because the arm has to do delicate movements to handle the wheel.

Although the arm is capable of a variety of movements, inserting a cross axel into a wheel - the most common way to build wheels - needs precision and force the arm is incapable of. This is why we decided to build a special wheel structure that the arm can handle easier. This structure is made of a cross axel, having the wheel at one end and a wedge belt wheel at the other end (see Figure 3). Such a wheel structure can easily be moved by the arm, by gripping the wheel.

It is worth mentioning that in case of real cars, changing the wheel does not require so much precision. You need precision to unscrew the bolts from the wheel, to put the spare wheel to the exact position where it is needed and to screw the bolts again. The problem of working with bolts could be solved by using some special mechanism, which makes them move (screw and unscrew) autonomously and they could somehow be attached to the wheel. However, you would still need an arm to take the flat wheel and put there the spare one.

Finally, the robot has a small platform built of Lego pieces in front of the arm, where the spare wheel is located.

3.2. **Software.**

The software of the robot is written in the C++ programming language, using the NXT++ library (available at [11]), an interface that allows controlling Lego Mindstorm robots through a USB or Bluetooth connection. Our choice was a connection through USB.

Moving the motors can be done in two different ways: using the *SetForward* method, which requires a connection to the brick, a motor and a value

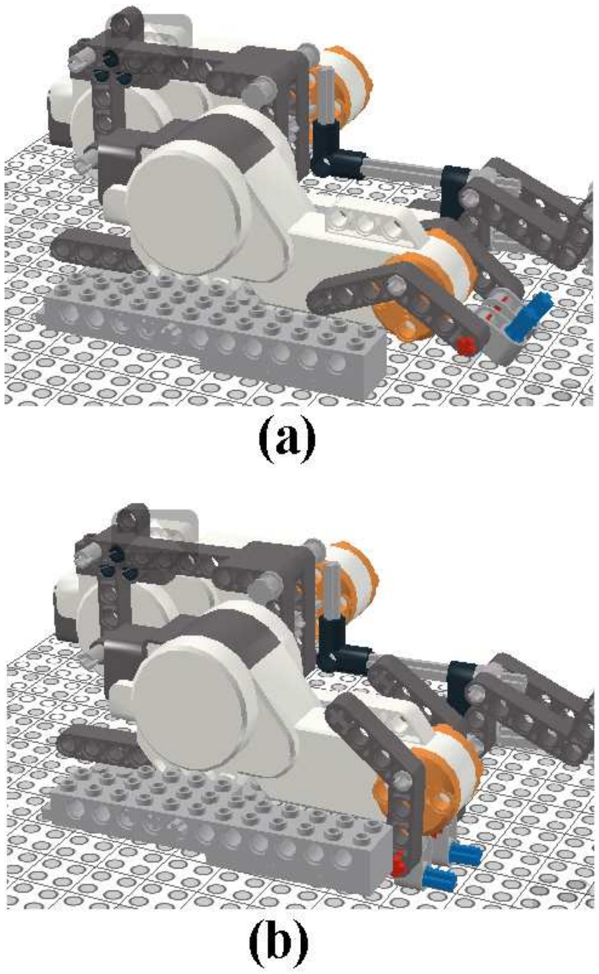

FIGURE 1. The mechanism for lifting the car. It consists of a motor which has attached a piece whose size is larger than the distance from ground to the chassis. When the car is down (a) it can move. When the car is lifted (b) it cannot be moved, but a robotic arm can replace its wheel.

between 0 and 100, the power. Using this method, the motor starts moving and continues to do so, until the *Stop* method is called for that motor. The second method is using the *GoTo* method, which requires a connection, a motor, the power, the degree to which the motor should be rotated and a Boolean value whether after the rotation braking should be used or not. When

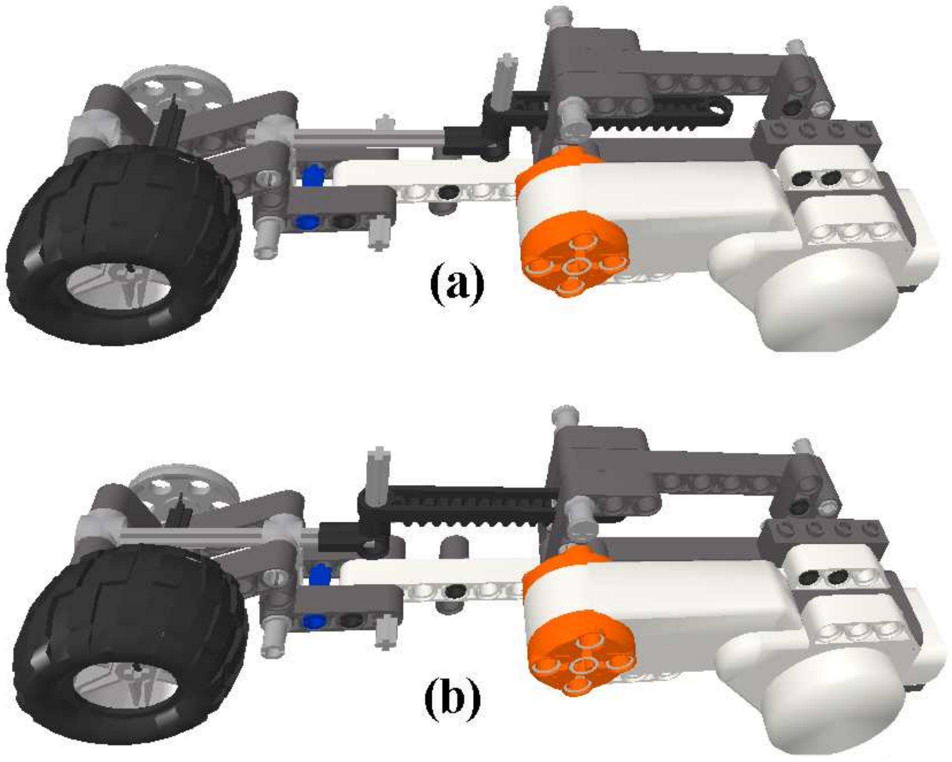

FIGURE 2. The mechanism for locking a wheel on the car. When the wheel is free (a) a robotic arm can replace it with a spare one. In this case the car cannot move because it risks losing its freed wheel. When the wheel is locked (b) the car can move.

the motor has turned to that degree, it stops, there is no need for the *Stop* method.

To control the Lynx arm we used again some simple methods to send data to and read data from the controller. Data sent to the arm is a string, starting with the "#" sign, followed by the number of the motor to be moved (from 0 to 5), a "P" followed by the pulse width in microseconds (500-2500), this shows how much the motor should move, and, finally, an "S" and the movement speed, which is optional, but here we used it to make slow movements.

3.3. **Detailed description of the procedure.** The robot goes forward for few seconds and then it stops to change the tire.

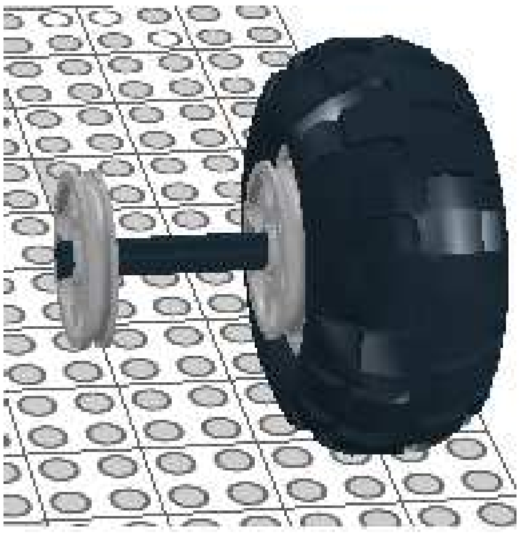

Figure 3. The structure of the wheel.

The first step is to reset the arm into a default position. This operation is not compulsory, but this helps us to simplify the movements of the arm.

Then the following steps are performed:

- The front part of the robot is lifted a little turning the second motor to 80 degrees as described in 3.1.
- This is followed by moving the axel to open the place where the wheel structure is located.
- Now the arm goes to the position where the spare tire is, gripes it, and slowly puts it down to the ground in front of the robot. It is important to move slowly, especially when the arm holds the wheel, because fast movements shake the robot, and the wheel might fall from the gripper. It is also important to put it down slowly, because the wheel, being round, might move from the place where it is placed, which is bad, because the robot has no camera to detect the wheel's position, so it looks for it where it was left. Fortunately, if the arm moves slowly, the wheel usually remains at the position where it was put to.
- The arm goes now and takes the left wheel structure, lifts it, and puts it to the platform where the spare tire was taken from.
- Then it goes back to the position where the spare tire was left, grips it and puts it to the place where it is needed. This is the most complicated movement, it is made of 11 fine movements of the arm to find the exact position.

- After this, the arm goes back to the default position. This position is the same as the one at the beginning of the algorithm.
- The axel is moved back, to close the place where the wheel is, the robot is put back on the ground and it moves forward for few more seconds, to see that it works the same way as at the beginning.

4. Experiments

No special environment is required for performing the experiments. The robot would perform the same on any type of hard floor.

In order to test the effectiveness of the procedure it was run 30 times. Out of these 30 experiments, 21 were successful. The most important reasons for failure are:

- the spare wheel put at the ground by the arm moved from the place where it was left. Thus, when trying to grip it again, the arm did not find it. It is worth mentioning, that if the wheel movement is a small one, than it is still possible that the gripper can get a hold of the wheel (as it can be seen on the movie that is mentioned at 6). This problem could be solved by using a camera to check if the arm is at the right position.
- when the axel was moved to lock it, it hit the axel of the wheel structure, and it was stopped by it. This led to the fall of the wheel structure when the robot started to move again. Although, we do not know why this happened, we think that making the front of the axel sharp would solve this problem.

The average time required for the arm to change the wheel is 2 minutes.

5. Strengths and weaknesses

In this section we describe the weak and strong points of our approach.

5.1. **Strengths.** Some advantages of the proposed approach are:

- The wheel is changed autonomously with no interference from the driver,
- The entire procedure is very simple and requires few minutes to perform,
- The algorithm is hard-coded and no complex pattern recognition algorithms [1] are required. Using a webcam and a vision algorithm for detecting the wheels would complicate the problem significantly [2, 3]. Fortunately in our case we do not need one.

5.2. **Weaknesses.** Some of the most important limitations of the system are:

- Cannot changes all its wheels. Changing the right wheel is the same as the case of the left wheel, but this operation requires an extra brick which will coordinate the fourth motor. Changing the other wheels is more complicated because we need a special mechanism which transmits the torque from motor to wheel.
- Cannot changes its wheels while driving. Doing this is simple if we change the wheels not attached to motors. In this case we simply need an extra place where to keep the flat wheel. Also, the mechanism which lifts the car should have a small wheel at its end. However, changing the wheels attached to the motors is very complicated because when the wheel is removed, no more torque is transmitted and the car would stop.
- The car requires an extra arm which is less likely to be placed on the real-world cars. The arm cannot be removed completely, but a smaller (specially designed) one could do the same job as the current one. Another option is to replace the wheel only in a specialized garage, but in this case the car should be able to run many kilometers with the tire flat. The good news is that there is a technology used by the GoodYear company, called RunOnFlat [12], which is based on the concept of reinforced side walls inside the tire, which keep the tire on rim and succeed in carrying the weight of the car for up to 80 km after a puncture.
- The entire procedure is currently run on an external laptop which is connected through cable to the robot. Placing the laptop on the robot is not viable. There are 2 alternatives here: one is to set a wireless communication between the robot and the laptop or to use a gumstix which is very small and can be placed inside the robot.
- A wheel from a real-world car has some brakes attached to it. In this case changing the wheel requires to detach the breaks, which is induces more complexity to the system.

We are working to fix all these limitations in a future version of the robot.

6. Conclusions and further work

Here we have described a self-repairing autonomous car which has the ability to change its wheels automatically.

The further efforts will be spent in the following directions:

- changing the wheel for motorized wheels,
- changing the wheels while driving,
- replacing other components of the car.

As recently the use of autonomous robots to ease the job of people in cars increased - for example in the DARPA Urban challenge [13] autonomous vehicles capable of driving without human help in traffic, performing complex maneuvers such as passing, parking and handling intersection are competing - the idea behind our robot seems to be a promising one, even if there still need to be different refinements (as mentioned at the above section at the disadvantages) to use this robot in real life.

Acknowledgement

A movie presenting an experiment can be seen here:
`http://www.youtube.com/watch?v=f1iHi8Pgb90`
This work was supported from grant IDEI-543/2007 from CNCSIS, Romania.

Mihai Oltean acknowledges the fruitful discussions on similar topics with the undergraduate students from the Intelligent Robots class of Faculty of Mathematics and Computer Science, Babeş-Bolyai University, Cluj-Napoca.

References

[1] Jain AK., Duin RPW., Mao J., Statistical pattern recognition: A review, IEEE Transactions on pattern analysis and machine intelligence, Vol 22, Issue 1, pp. 4-37, 2000
[2] Jain R., Kasturi R., Schunck B.G., Machine Vision, McGraw-Hill, 1995
[3] Forsyth DA., Ponce J., Computer vision: a modern approach, Prentice Hall, 2002
[4] Zykov V., Mytilinaios E., Adams B., Lipson H., Self-reproducing machines, Nature Vol. 435 No. 7038, pp. 163-164, 2005
[5] Bongard J., Zykov V., Lipson H., Resilient Machines Through Continuous Self-Modeling, Science Vol. 314. no. 5802, pp. 1118 - 1121, 2006
[6] Robotic Chair: www.news.cornell.edustoriesOct06robotic.chair.aj.html
[7] Robotic Chair: www.raffaello.nameRoboticSculpture.html
[8] Gumstix: Way Small Computing, www.gumstix.com
[9] Modular Robot: www.nytimes.cominteractive20090727science20090721-modular-graphic.html
[10] Lynx Arm: www.lynxmotion.comCategory.aspx?CategoryID=25
[11] NXT++: nxtpp.clustur.com
[12] RunOnFlat: eu.goodyear.comhome_entiresrunonflat
[13] DARPA Urban Challenge: http:www.darpa.milgrandchallengeindex.asp

Department of Computer Science, Faculty of Mathematics and Computer Science, Babeş-Bolyai University, Kogălniceanu 1, Cluj-Napoca, 400084, Romania

E-mail address: `sdsd0092@scs.ubbcluj.ro`
E-mail address: `mzsi0142@scs.ubbcluj.ro`
E-mail address: `moltean@cs.ubbcluj.ro`