Robotour Solution as a Learned Behavior Based on Artificial Neural Networks

Miroslav Nadhajský and Pavel Petrovič
Department of Applied Informatics, Faculty of Mathematics, Physics and Informatics, Comenius University,
Mlynská dolina, 842 48 Bratislava, Slovakia, miroslav.nadhajsky@st.fmph.uniba.sk, ppetrovic@acm.org,

Abstract—**Our contribution describes a mobile robot platform that has been built for the purpose of the contest Robotour – robotika.cz outdoor delivery challenge. The robot is a standard differential-drive robot with a good quality consumer market digital video camera with a lightweight, but high-performance laptop computer used as the main control board. Supplementary board is used to control motors and sensors of the robot. The robot utilizes a behavior-based architecture and its vision module that is responsible for track-following is utilizing an artificial neural network that was trained on a set of images. This is a novel solution that has not been used in Robotour contest previously, and our early experiments demonstrate promising results.**

Keywords – robotour, navigation, artificial neural networks, learning robots

I. Introduction

Applications of robotics technology in both production and personal use are becoming possible with the development of new materials, motors, sensors and vision, ever decreasing cost of computing and memory capacity, and development of new algorithms and control strategies. Robots must be able to operate in dynamic and unpredictable environments. Therefore, one of the most important challenges to be solved reliably is robot navigation – in both indoor and outdoor environments. The robots must be able to localize themselves on a supplied map, create their own map representations of the explored environment, and they must be able to navigate their environments safely, without colliding with obstacles, or failing to follow the paths, roads, trails, and tracks. The real improvements in the technology typically occur when there is a large motivational pressure to produce a working solution. This might either be a goal to produce a final product, or alternately, with somewhat more relaxed requirements and settings, which are suitable for experimentation, and research, when the goal is to develop a robot to participate in a robotics contest.

Robotour – robotika.cz outdoor delivery challenge, organized by the Czech association robotika.cz, is an annual meeting of teams building and/or programming outdoor robots that navigate in a city park filled with trails, trees, grass, benches, statues, water ponds, bridges, and people. The task changes every year, but the main challenges are 1) be able to localize and navigate on a map supplied by the organizers, and 2) be able to follow the trails and paths without colliding with the obstacles or leaving the path without reaching the goal. See [1] for the exact rules of this year's contest.

Various solutions for the challenge were developed, however, in most cases, they did not take advantage of advanced artificial intelligence algorithms. In particular, only few different vision algorithms were developed until today, several teams shared the successful solution of [2], and many solutions rely on the use of odometry, compass, and GPS. We would like to address this area, and prepare a solution for the contest in 2010 or 2011 that will utilize AI algorithms. The second author has participated in the competition team several times in the past, and collected some experience and motivation for a new attempt. In this article, we describe the principles our solution is based on and is currently being built. In the following sections, we describe the mechanics and the hardware, robot overall architecture, the software components, and the AI methods that we aim to use. Finally we summarize the experience with building and programming the robot up to date.

II. Mechanics

The robot is a simple robot with differential-drive kinematics with one supporting free-rolling caster wheel. The length of the sides of its square base is 45 cm; the air-inflated wheels of a diameter 15.3 cm are mounted on the outside of the base, in the front of the robot. The total weight is about 6 kg without any load. The robot provides a storage space of ca. 20 x 20 x 45 cm to carry a heavy load (approx. 5 kg), which can be placed close to the center of rotation, above the propelled wheels, so that it does not have a negative impact on maneuverability of the robot. The main control unit is a portable computer, mounted in a flat plastic frame with a foam to compensate the shocks. The lead acid 12V 9Ah rechargeable battery, being the heaviest component, is stored under the base, between the wheels, keeping the centre of gravity low. Color camera with a true optical image stabilizer and CCD image sensor is mounted using anti-shock foam on a U-shape construction frame built of aluminum profiles, together with GPS and IMU sensor, see Fig.1. The camera is inclined 10° downwards. The IMU sensor must be mounted far from any sources of electric and magnetic fields, such as motors and wires. Placing GPS high compensates also for obstacles in the surrounding terrain, which may hinder the GPS satellites signal. The robot is built from raw materials, except of the motors, wheels and consoles that hold them, which are all part of a set from Parallax. The aluminium framework allows mounting a rain shield for the computer and the camera when necessary.

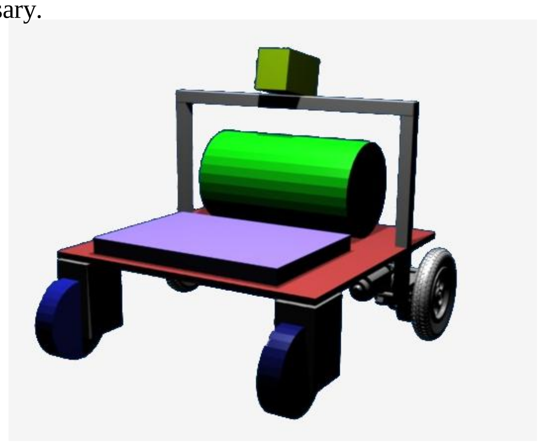

Figure 1. 3D Model of the robot showing main parts. In real implementation, we have mounted only one caster wheel as it proved to be sufficient, and allowed more accurate control.

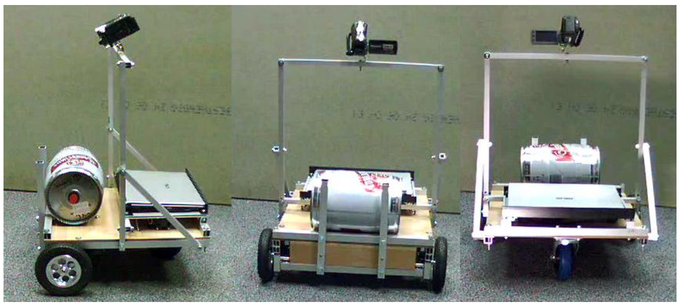

Figure 2. The resulting constructed robot from the side, front, and back. The control electronics is installed under the PC. The robot has already been tested in outdoor settings and has traveled a distance of several km.

III. Hardware Architecture

The robot is propelled by two 12V DC motors with built-in transmission, rotating at up to 150 rpm and consuming 1.5A at no load. The encoders with 36 ticks per rotation are used for speed and position feedback and are equipped with on-board microcontrollers that are directly connected to the motor drivers HB25, supplying them with the proper PWM signal to keep the requested speed. In this way, the main microcontroller board, which is the SBot control board, designed in our group originally for SBot mobile robot, is freed from the low-level motor control, and dedicates this task to both of the encoders that have an implementation of a standard P (proportional) controller and are connected using the same 1-wire serial bus. Unfortunately, we found that the original firmware for the encoders supplied by Parallax did not satisfy our needs for several reasons. Most importantly, the encoders were not designed for dynamic change of speed, but only for simple positional commands that accelerate from zero speed to a fixed predefined speed, and then decelerate after traveling the required distance. They do not allow to change the speed in the middle of such positional command. However, movements, where the speed and rotation is changed arbitrarily at any time, are required in the Robotour task, where the robot has to dynamically respond to the visual feedback when it has to align its movement with the shape of the path. Fortunately, Parallax makes the source-code for the encoders firmware available, and thus we could modify it to suit our application and support immediate smooth changes of the instant speed.

The obstacles are detected using the standard SRF-08 and Maxbotix LV EZ1 ultrasonic distance sensors that are connected to the main control board.

Outdoor robots are typically equipped with a global positioning device, i.e. GPS, and it is the case for our robot too. Information from the GPS module that is connected directly to the main computer using USB port, however, is not so reliable due to atmospheric and other occlusions, and serves only as a guidance for map localization. It is confronted with visual input and complemented by the current heading obtained from compass sensor. The compass sensor is part of the complex 9 DOF IMU sensor that includes several axes of gyroscopes, accelerometers, and magnetometers, thus compensating for various robot inclinations when traveling uphill or downhill. This is important since the simple compass sensors provide incorrect information once the robot and thus also the sensor is tilted.

Finally, for the visual input, we chose to use a standard video camera Panasonic SDR-T50, due to a very good ratio of parameters/price. The video camera is built around a CCD sensor, which has the advantage over the CMOS image sensors of taking the image instantly. Cheap CMOS cameras therefore suffer from a serious vertical distortion when the camera is moving, since the different rows of the image are scanned at different times. In addition, the camera has a built-in true optical image stabilizer, which further compensates for distortions due to the movement. Unfortunately, we found this stabilizer to be insufficient, and thus we have supported it with an anti-shock foam placed between the camera and the platform where it is tightened using flexible textile tape. The camera renders its image either as 16:9 or 4:3 image, however, it sends a wider signal down to its video output jack connector, which is further connected to a USB frame grabber card and the main computer. The main computer is a 2-core powerful PC with a GPU that can be used for the intensive image processing computation. The computer and the Sbot control board are connected using a serial port or a virtual serial port over radio BlueTooth connection. In debugging and testing applications, the robot can be controlled using a wireless gamepad connected using a proprietary 2.4GHz radio link.

In general, the robot is designed in such a way that it can be used in many different applications. For instance, a stereo vision system or an arm with a gripper can be installed in the cargo hold area. Additional sensors can be easily mounted on the aluminum profiles or wooden base. Fig. 3 shows overall system architecture.

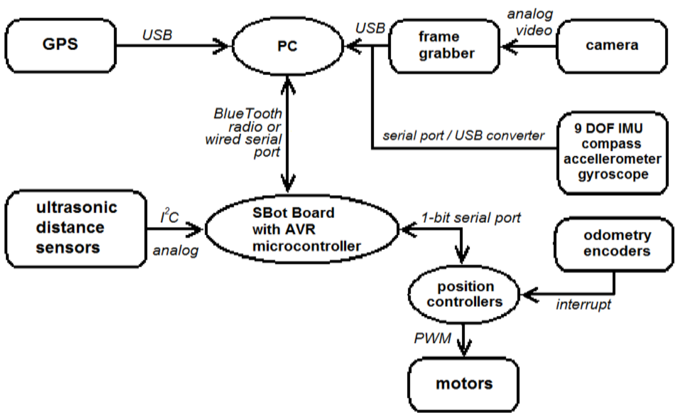
GPS
USB
PC
USB
frame grabber
analog video
camera
BlueTooth radio or wired serial port
serial port / USB converter
9 DOF IMU compass accellerometer gyroscope
ultrasonic distance sensors
I2C
analog
SBot Board with AVR microcontroller
1-bit serial port
odometry encoders
position controllers
interrupt
PWM
motors

Figure 3. System hardware architecture.

IV. Software Controller Architecture

The software architecture is tailored for the Robotour contest. In this year's contest, the goal for the robot is to navigate to the target without knowing its starting location. It is only given the target coordinates and an official map of the park. It may not use other map information. The software controller is logically divided into five main components, see Fig.4.

The first component, planning, uses the map with the destination location and generates a path plan for the robot to follow. It tries to minimize the number and complexity of the crossings as these are the most critical places and candidates for navigational errors. The component outputs a sequence of locations that are to be visited by the robot. Whenever requested, the module can generate a new plan after a problematic place in the map has been reached.

The second component, localization using map, is responsible for the most accurate localization of the robot on the map. It is using the information from the compensated compass (IMU) for heading, from GPS for position estimation, and from the position encoders to estimate the distance traveled and turns made. All the information is integrated and with the help of the map and the path plan, the target distribution is determined using a probabilistic Monte-Carlo estimation. The output of the localization module is a probabilistic distribution over the expected heading in the very next correct movement, and the expected distance to the next crossing or target.

The third module, path recognition, is the most important one for the actual control of the motors, and has a priority over the localization module. It receives the image from the front camera and recognizes which parts of the image correspond to the path, and which of them correspond to other surfaces. The next section explains this procedure in more details. The output of this module is again a probabilistic distribution over the space of possible headings that can be projected to the input frame, where the headings leading to more "path" areas are more likely than those leading to less "path" area. Input from the odometry and gyroscopes helps this module to improve its estimation of the path using its previous estimations and the relative displacement of the robot.

The obstacle recognition module is responsible for detecting obstacles in the planned path of the robot and for stopping the robot in case of a possible collision early enough so that avoidance could be attempted by the coordination module. The robot is currently equipped with three ultrasonic distance sensors (front ahead, front left, front right), and thus the module reports on its output whether the path is blocked completely, or only partially, and also what is the size of the expected free buffer in front of the robot.

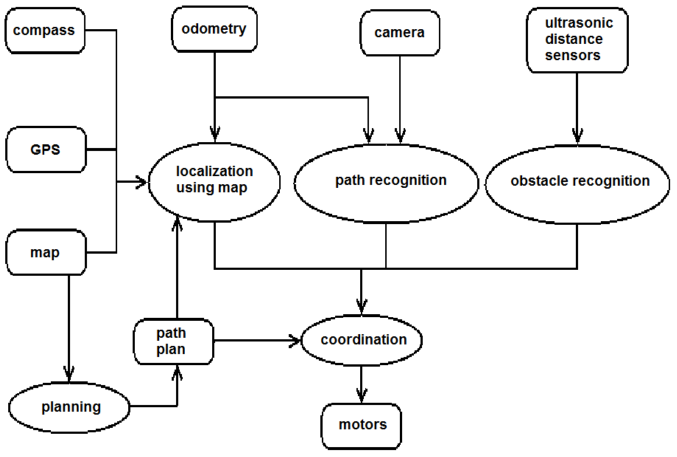

Figure 4. Overall controller architecture.

The most complex module is the coordination module. Its purpose is to take the prioritized outputs from the other three modules, and to determine the best possible angular and linear velocity for the next instant movement. When the confidence of the module is getting low, the robot slows down. If the confidence falls even lower, the robot stops, and starts rotating left or right, depending, which direction is expected to be more promising, until it finds a heading, where the module confidence is sufficiently high again. If such heading is not found, the robot attempts to return back in the reverse direction as it arrived to the problematic location, possibly moving in the reverse of the planned direction on the map. After returning back a short distance, it retries. The retries are repeated several times while gradually extending the back-up distance. If all attempts to pass the problematic location fail, the planning module is asked to generate a different path.

The controller is arranged in a behavior-based manner, individual behaviors are developed and tested independently before they are integrated in a common controller.

V. Path Recognition

Our goal was to use artificial neural networks in order to help the robot navigate and stay on the path. We obtained many images from a park with trails, and we have manually marked the regions in these images that correspond to the traversable path. This input was used to train the neural network (a standard multi-layer perceptron) to recognize the path. See figure 5 for an example of such manually classified image.

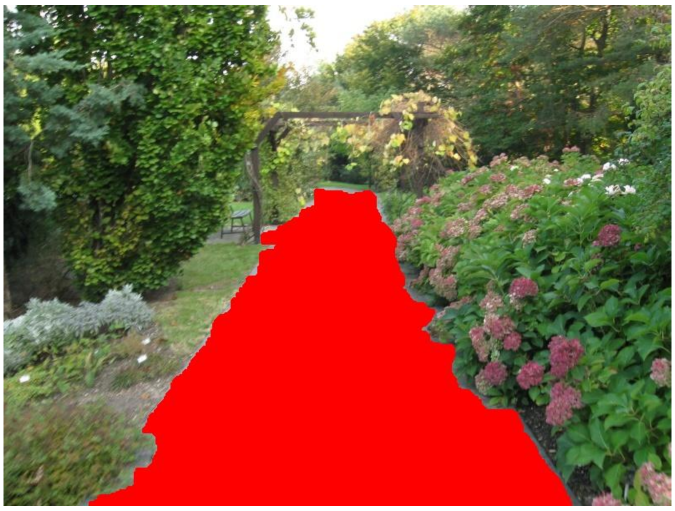

Figure 5. Manual preparation of training images.

Sending the whole image to the network as the input would obviously be infeasible. Instead, we first tried to scale the image to a lower resolution of 400x300 pixels, and divide it into 100 rectangular regions of equal sizes that covered the whole image. Each region formed an input to a neural network, and the whole region was about to be classified as "path" or "not path". However, the resulting resolution of the classified image was not satisfactory, even after a further reduction of the region size so that the image was divided into 2500 segments. Therefore, we decided to use a sliding region. For almost every pixel in the image, we define a corresponding region – it's larger neighborhood, which forms the input vector. The classification output produced by the network for each pixel in the image is then a real number from 0 to 1, estimating how much the network believes the pixel lies on the path. Two examples of images that were not used in the training phase are shown in the Fig.6.

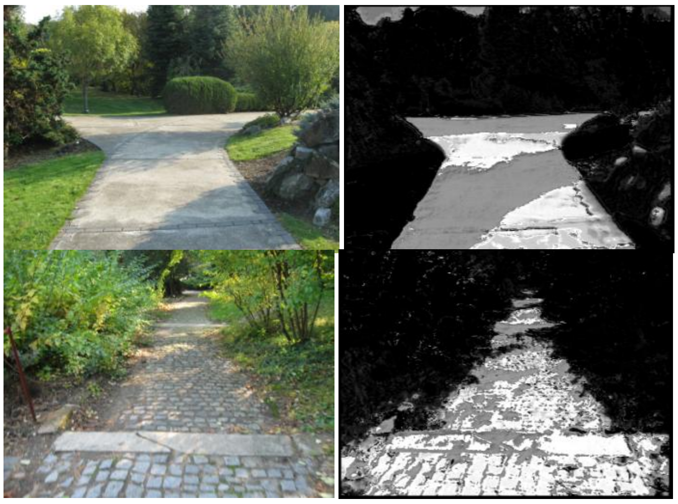

Figure 6. Examples of path recognition.

We used the RPROP training algorithm for multilayer perceptron, in particular the implementation that is present in the OpenCV package. The training used tens to hundreds of manually classified images from various places in a park with various path surfaces, light and shadow conditions. Since this is still an ongoing work and only preliminary results are available, we restrain from a statistical analysis of the results at this moment, and refer the reader to the page dedicated to the project with detailed results and data [5].

Once the network is trained and produces the classifications for the image frame pixels, the path recognition module enters a second phase, when it tries to evaluate all possible travel directions (headings) with respect to the chances that the robot will stay on the path. For this purpose, the module analyzes a family of triangles of the same area with the base at the bottom of the frame and the third vertex placed in the middle of the image. For each such triangle, we compute an average path likelihood. The triangle for which the path is most likely, i.e. where most pixels lay on the path, is likely to be the correct new heading. However, the module outputs a full distribution over all possible headings so that the coordination module can take advantage of this information, for instance to determine different directions at a heading, or when trying to resolve ambiguous cases. Fig.7 depicts the analyzed family of triangles. Two example pictures are further analyzed in Fig. 8, where the bars show how "likely" it is that following in the various directions is a "good" idea in order for the robot not to leave the path.

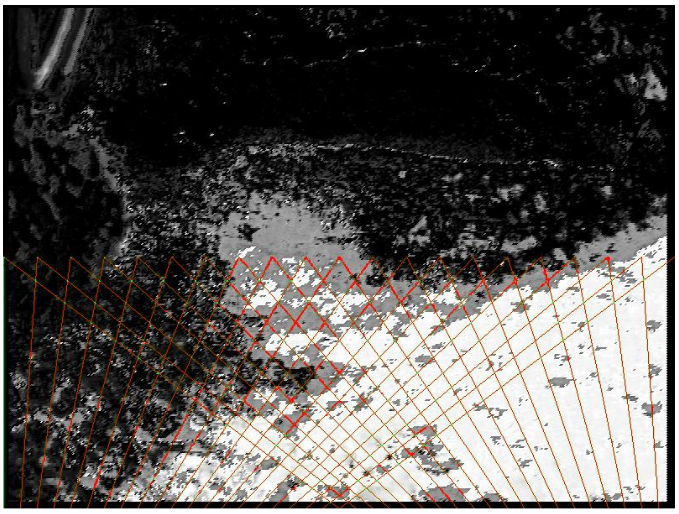

Figure 7. Triangles representing different turning projected to the image of recognized path.

VI. Conclusions and Future Work

We have designed and implemented a robotic hardware and software platform to be used in the Robotour contest for outdoor robots navigating in park environment. The hardware platform is implemented in a general way and most components of the software platform can be reused in other applications, the robot can be extended with stereo vision or manipulator. We have designed, implemented and tested in this context a new method for path recognition, which is based on artificial neural network that is trained on a set of static images that are similar to the environment where the robot is to be operating. We are currently working on integrating all the components of our prototype so that it could perform in its first

Robotour contest this year. In the remaining 10 months of the project, we will analyze the results from our participation, and propose, implement, and verify improvements so that the robot can serve both as a competitive platform in the contest and as an educational tool in the course Algorithms for AI Robotics, which is provided at our department to students of Applied Informatics.

Acknowledgment

This work has been done with the support of the grant "Podpora kvality vzdelávania na vysokých školách" from Nadacia Tatra banky.

References

[1] Robotour - robotika.cz outdoor delivery challenge, Rules, online: http://robotika.cz/competitions/robotour/cs last accessed: August 1st 2010.

[2] K. Košnar, T. Krajník, and L. Přeučil, "Visual Topological Mapping," in European Robotics Symposium 2008, Heidelberg: Springer, 2008, pp. 333–342.

[3] G. Bradsky, A. Kaehler, Learnning OpenCV: Computer Vision with the OpenCV Library, O'Reilly Media, Inc., 2008.

[4] D. Guštafík, SBot v2.0 – Educational Robot for Clubs and Classrooms, User manual to Sbot robot, 2008, online: http://webcvs.robotika.sk/cgi-bin/cvsweb/~checkout~/robotika/sbot/2.0/doc/dokumentacia_en.pdf

[5] M. Nadhajský, Robotour diploma project page, 2010, online: http://virtuallab.kar.elf.stuba.sk/~mnadhajsky/

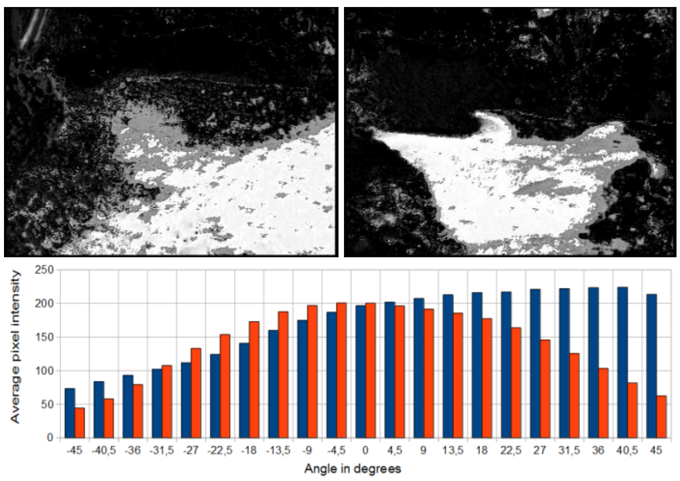

Figure 8. Two scenes after path recognition. The bars show the average pixel intensity of pixels inside of triangles for a range of different rotations for both of the resulting images (blue/dark for the left image, red/bright for the right image).