FetchBot: developing an autonomous fetch robot companion

Marianne Bossema, Joost Mollen, Jordy van Miltenburg, Martijn Wester
Project paper for the course "Robotics"
Leiden Institute of Advanced Computer Science, Leiden University, The Netherlands
April 2020

***Abstract*— This paper presents the efforts of a three-month study to create a robotic companion with autonomous fetch behaviour called FetchBot. The design, implementation, challenges and results are discussed.**

I. INTRODUCTION

An active effort is made to transforms robots from tools into social entities (Samani, 2013). Specifically, there has been an effort to create robot companions that would potentially fulfil the same domestic niche as pets. Examples include Sony's Aibo (Sony, n.d.), Anki's Vector (Anki Vector, n.d.) or Paro the seal (Wada et al., 2010), which are designed to entertain or function as a therapeutic tool much akin to animal therapy, without actual animal care (Petersen 2017). Artificial pets offer an interesting platform to study the "morphological and behavioural characteristics that stimulate interaction and elicit meaning attribution and attachment from the human beings" (Marti, 2005).

We aim to design a robot with which a user can have a playful interaction in a domestic setting. Specifically, we focus on the activity of playing fetch: a pet picks up an object a human throws away. The robot should continuously and autonomously search for a ball in a stable environment, pick it up, throw it away and repeat this process. This allows for a setting in which the bot is able to play both with itself as with a human. We call this robot: FetchBot.

Fetch behaviour in robots is not novel. Warehouse robots that are able to locate and pick-up items are getting increasingly common (Bogue, 2016). Focussing on the entertainment industry, researchers at the UCSD Coordinated Robotics Lab have developed a radio-controlled robot called iFling that is able to pick-up, store and throw a ping pong ball (Chen, 2010). The novelty that FetchBot introduces is that it is an *autonomous* fetch robot aimed for *domestic* use. Furthermore, FetchBot could function as platform to study human-robot interaction and robot search behaviour.

II. METHOD

Design

To develop FetchBot, we employed a modular strategy, developing the components of the system separately. This allowed us to work on features in parallel. We organized several brainstorm sessions to discuss possible solutions for these components and how to integrate them. The requirements for FetchBot concern several capabilities, namely locomotion, picking up and launching a ball, and autonomous behaviour that allows for searching and finding a ball in a controlled environment using colour, blob and line detection.

As a basis for our prototype and to enable locomotion we used the Monsterborg (MonsterBorg, n.d.), which can be programmed as a self-driving robot and can be extended with a camera, extra circuit boards, sensors and servos. In our case, the Monsterborg was equipped with a Raspberry Pi 3 Model B+ board and a Raspberry Pi camera. We also used a smaller Yetiborg (Yetiborg, n.d.) - equipped with a Raspberry Pi Zero board and a Raspberry Pi camera in order to prototype and test the search, track and line detection software.

Implementation

The components we added to the Monsterborg were developed in the following manner.

Search mechanism

To enable FetchBot to search a ball, we initially incorporated it with a search algorithm inspired on existing animal foraging techniques. Research in biology on dynamic search patterns hypothesise

that - given an environment with a low target density and scarce information - similar optimal foraging strategies will emerge that can be modelled with a Lévy walk: a random walk with an occasional long step (Wosniack, 2017). Examples of search behaviour that follow Lévy walk foraging strategies include the foraging patterns of birds and hunter-gathering tribes in Namibia and Tanzania (Wosniack, 2017).

However, the Lévy-walk inspired foraging technique did not outperform a random strategy in which Fetchbot moves straight and when encountering a line, chooses a random new angle to follow. When the input angle is shallow Fetchbot will make a small change in contrast to when the input angle is 90 degrees. Then the output angle is bigger and more random. There are several possible explanations as to why the Lévy walk foraging model did not outperform our random search strategy, which are discussed in the Discussion section. For a search strategy, we, therefore, settled eventually on a simple strategy: Fetchbot drives straight until it either detects a ball or a line. When detecting a ball, it drives towards it. When it detects a line, it picks a random angle in which to drive away from the line.

Ball detection & Line Detection

The Yetiborg and Monsterborg used were equipped with a Raspberry Pi camera. For both the ball and line detection, we used blob detection with OpenCV (OpenCV, n.d.) in combination with a custom made tool for easy HSV colour detection (Fig .1). This stand-alone tool takes pre-recorded videos and is able to extract optimized HSV bounds. It is written in C++ making use of OpenCV and OpenFrameworks. By dragging the mouse over the desired object's color, the program automatically determines the HSV bounds that are later used for color detection, and is optimized to ignore noise.

The obtained values are used in blob detection to detect the ball. This ensures that FetchBot would only pick up a ball of a predetermined colour. For example, FetchBot should search for an orange ball and ignore a green ball. Furthermore, to avoid the detection of small discrepancies, the program also calculates the area of the detected blob so only blobs of a particular size are considered. For line detection, the centre point of the detected blob is calculated in order to figure out in which direction Fetchbot should turn to make a more efficient turn.

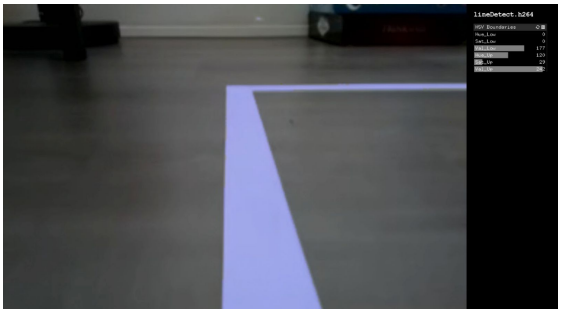

Fig 1. Custom tool to select HSV colour ranges

Locomotion

The MonsterBorg can drive autonomously based on images from the camera. Those images are processed using a Python script, running on the Raspberry Pi. OpenCV colour detection is used to track the ball, and the coloured lines that frame the area of our controlled environment. Once a coloured line is found in front of the camera, the Monsterborg rotates and moves further in a randomly chosen direction. Once the ball is detected, the Monsterborg proceeds with picking up and launching the ball, as described below.

Picking up mechanism

We discussed and experimented with different grabbing mechanisms (Fig. 2). We decided that the mechanism would be most reliable with few degrees of freedom, so that there wouldn't be much dependencies between actuators. After some prototypes, we came up with a shape that allowed us to catch and pick up the ball and place it into the launching tube in one motion (Fig. 3). The angle and length of the arm are tuned to move the ball upwards when tilted, placing it in the tube-shaped launching mechanism. The grabber was created out of a plastic card, connected with a small arm to a servo motor which we mounted in the front of the Monsterborg, below the camera (Fig. 4).

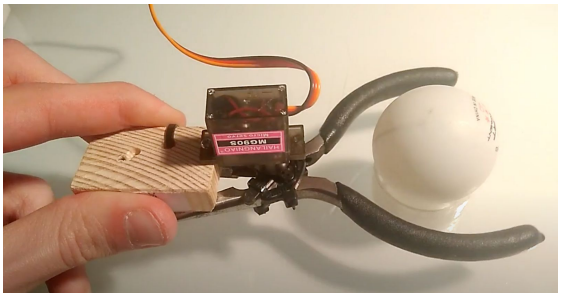

Fig 2. Exploring grabbing mechanisms

For the ball detection, a combination of an infrared LED and an infrared sensor is used. These components are placed opposite to each other on both sides around the ball catching hole. About 20 times per second, an infrared light pulse is emitted by the LED and captured by the sensor on the other side. When a ball intercepts the beam, the lifting action is triggered. When the arm is in front of the launching tube, it is checked once again whether the ball has indeed been successfully placed. If so, the arm is lowered and the launching sequence starts. The hardware for picking up the ball is controlled by an Arduino script.

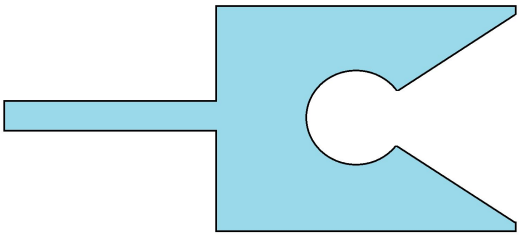

Fig 3. Shape of the lifting arm , top view.

Launching mechanism

For launching the ball we used a 12V DC solenoid mounted inside a red plastic tube, which was originally a medicine bottle (Fig. 4). The solenoid is used to push against the ball in order to shoot it. To shoot with enough force, 12V would certainly be insufficient. For a decent launch, a higher voltage in combination with a large current was needed. To achieve this kind of electron flow on battery power, we used two 200V capacitors of 1000µF each and mounted those on the Monsterborg. A step up high voltage booster board was needed to crank up the voltage. This way, all components could run on the same Monster borg battery pack.The launch sequence starts with charging the capacitors, and when full, it releases all energy at the solenoid. The switching of these currents is done with two relays, compatible with Arduino (5V).

We also did several experiments to explore the effects of the force generated by the solenoid, and tested different projectiles. Although being classified as suited for 12V continuous, the solenoid was able to handle the large current spikes without problem. Even shooting with 300V (with other capacitors: 2x 560µF) was no problem, presumably because of the short duration of the current flows. Nevertheless, for our project, to keep the ball in bounds of the controlled environment, we used around 120V.

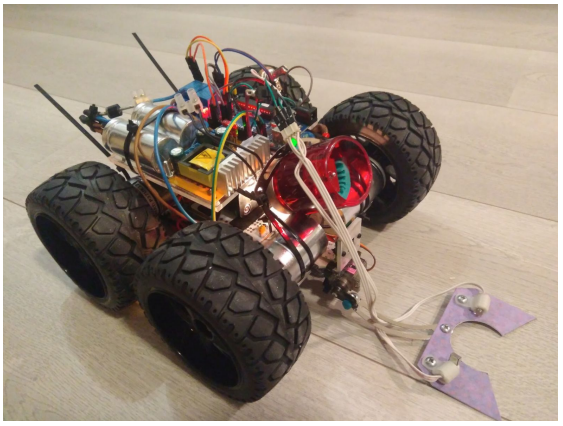

Fig 4. Monsterborg with lifting/launching features

III. RESULTS

The results of individual components matched our expectations. The Monsterborg was equipped with mechanism which could successfully pick-up and shoot a ping pong ball. The software for dynamic search and line detection did also work as expected in the Yetiborg. These results can be observed in the provided video material.

However, the integration of several individual components proved to be challenging. The robots behaviour software, successfully tested on a Yetiborg, appeared to be problematic once run on the Monsterborg. More time and research is needed to integrate these two components. We reflect on this challenge in the chapter Discussion.

IV. DISCUSSION

Although it seemed a good approach to work on the different components simultaneously, this eventually brought up issues of bringing the parts together. Because the software for detecting the ball and lines was developed on the Yetiborg, the code had to be ported to the Monsterborg which was not as straightforward as expected. It was hard to collaborate on these problems. It would have been a better approach to first set up an umbrella project, work out a plan for integration and/or test

small pieces of the software on both Yetiborg and Monsterborg.

Concerning the search algorithm, there are several possible explanations as to why the Lévy walk foraging model did not outperform our random strategy. While Fetchbot did have relatively scarce information - a fixed Pi camera video feed - the test environment was to small, resulting in no noticeable performance difference on when the ball was detected. This technique could be revisited when in future research a larger test area is used. Another reason why a Lévy walk foraging strategy might have underperformed to expectations was that employing a series of quick turns might have given Fetchbot not enough time to detect a ball in its video feed.

Future research

There are several components of this study that would benefit from future research. First and foremost, it seems essential to implore a more integrated implementation strategy in future research. While individual components functioned as expected, implementation ran into a challenge integrating the behaviour software (tested successfully on a Yetiborg) on the Monsterborg with the pick-up and launch mechanism. This caused our end result to divert from the original goal.

Several additions could also be made to FetchBot in future research such as face recognition which would allow FetchBot to shoot the ball to a (specific) human and improved object detection to be able to avoid objects and operate in a realistic domestic setting. When FetchBot would be tested in a larger environment, the potential relevance of using a Lévy walk foraging strategy could also be revisited. Lastly, it would also be interesting to research FetchBots in the public space.

V. CONCLUSION

We aimed to design a robot that can move around autonomously and can continuously search for a ball, pick it up and launch it: a FetchBot. We partly achieved our goal, by implementing the different components that were required. However, to integrate these features into a fully functional autonomous robot that allows for playful human-robot interaction, more time and effort is required. We suggest that, when choosing a strategy of modular development of robotic parts that are developed standalone and in parallel, it is essential to anticipate on future issues of integration already in an early stage. Nevertheless, as a prototype, and to explore possible implementations for the different features, our FetchBot matched our expectations.

REFERENCES

Anki Vector. (n.d.). Retrieved June 1, 2020, from <https://anki.com/en-us/vector.html>

Bogue, R. (2016). Growth in e-commerce boosts innovation in the warehouse robot market. *Industrial Robot: An International Journal*.

Chen, P. T. (2010). Simulation and optimization of a two-wheeled, ball-flinging robot (Doctoral dissertation, UC San Diego).

Marti, P., Pollini, A., Rullo, A., & Shibata, T. (2005, October). Engaging with artificial pets. In *ACM International Conference Proceeding Series* (Vol. 132, pp. 99-106).

MonsterBorg - The Ultimate Raspberry Pi Robot. (n.d.). Retrieved May 27, 2020, from <https://www.piborg.org/robots-1/monsterborg>

OpenCV. (n.d.). OpenCV. Retrieved May 27, 2020, from <https://opencv.org/>

Petersen, S., Houston, S., Qin, H., Tague, C., & Studley, J. (2017). The utilization of robotic pets in dementia care. *Journal of Alzheimer's Disease*, *55*(2), 569-574.

Samani, H., Saadatian, E., Pang, N., Polydorou, D., Fernando, O. N. N., Nakatsu, R., & Koh, J. T. K. V. (2013). Cultural robotics: the culture of robotics and robotics in culture. *International Journal of Advanced Robotic Systems*, *10*(12), 400.

Sony. (n.d.). AIBO unleash wonder. Retrieved June 1, 2020, from <https://us.aibo.com/>

Wada, K., Ikeda, Y., Inoue, K., & Uehara, R. (2010, September). Development and preliminary evaluation of a caregiver's manual for robot therapy using the therapeutic seal robot Paro. In *19th International Symposium in Robot and Human Interactive Communication* (pp. 533-538). IEEE.

Wosniack, M. E., Santos, M. C., Raposo, E. P., Viswanathan, G. M., & da Luz, M. G. (2017). The evolutionary origins of Lévy walk foraging. *PLoS computational biology*, *13*(10), e1005774.

YetiBorg. (n.d.). Retrieved May 27, 2020, from <https://www.piborg.org/robots-1/yetiborg-v2>