1 2 90

UNIVERSIDADE D
COIMBRA

Artur João Anjos de Oliveira

Ultrasound Tracking and Closed-Loop Control of Magnetically-Actuated Biomimetic Soft Robot

Dissertation for Integrated Master's Degree in Biomedical Engineering under the supervision of Professor Jorge Batista and Professor Venkat Kalpathy Venkiteswaran and presented to the University of Coimbra

February, 2022

University of Coimbra
Faculty of Sciences and Technology

Closed-loop control of magnetically-actuated soft robots with imaging feedback

Artur João Anjos de Oliveira

Dissertation presented to the Department of Physics of the Faculty of Sciences and Technology of the University of Coimbra

Supervisors:
Professor Doctor Jorge Batista (DEEC/UC)
Professor Doctor Venkat K. Venkiteswaran (DBE/UT)

Coimbra, 2022

1 2 9 0

UNIVERSIDADE D
COIMBRA

O presente trabalho foi realizado com a colaboração de um grupo internacional e multidisciplinar de investigadores e foi submetido para publicação na conferência: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

The present work was accomplished through the collaboration of an international and multidisciplinary group of people and is currently being reviewed to publishing in the conference: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Abstract

Untethered magnetically actuated soft robots can provide potential medical applications and revolutionize the field of minimally invasive interventions. Its soft, untethered nature allows the navigation to difficult-to-reach targets of the human body without damaging the surrounding tissues. Moreover, magnetic actuation is radiation-free, not harmful for humans and removes the need to have an on-board source of energy in the robot. Despite the recent developments in the design and actuation of soft robots, there are some challenges, such as localization, perception, and path planning, to overcome so that they can perform tasks in challenging environments.

The main goal of the current project is to achieve closed-loop motion control and planning of a soft robot, the Millipede, using ultrasound imaging technique. In this study, we integrate localization and control strategies into a magnetic actuation system to safely steer the untethered soft robot to a target. Regarding the control, a Proportional Integrative (PI) controller is used to calculate the linear and angular velocities to steer the robot through the workspace while avoiding obstacles. According to the velocities, the corresponding magnetic field is applied, using a setup with six electromagnetic coils. The localization is first obtained from a top-view camera as a proof-of-concept of the motion control and planning methods. Then, we compare the performance between two ultrasound algorithms, geometric and a deep learning approach, to estimate the pose of the Millipede. Finally, the closed-loop control of the untethered soft robot is achieved using ultrasound imaging.

The results show the possibility of using the soft robots to autonomously perform tasks in clinically relevant scenarios. The proof-of-concept experiment with the top-view camera imaging of the workspace achieved a deviation error between the Millipede position and the path planned of 1.71 ± 1.07 mm. The geometric and deep learning approaches obtained an average tracking error for the position of 0.68 ± 0.50 mm and 1.5 ± 2.5 mm, respectively, and for the orientation an error of 2.3 ± 3.4°, and 3.9± 6.7°, respectively, on a dataset with 9500 ultrasound images. Finally, the closed-loop control of the Millipede, using the deep learning tracking in the ultrasound images, showed a tracking error of 1.6 ± 0.84 mm and an angle error of 2.23 ± 1.48 mm. The deviation of the robot from the path planned was 1.87 ± 0.92 mm.

Keywords:
Minimal Invasive Interventions, Untethered Soft Robots, Magnetic Actuation, Motion Control, Tracking, Deep Learning, Ultrasound Imaging

Resumo

Soft robots atuados por magnetismo podem fornecer potenciais aplicações médicas e revolucionar a área de intervenções minimamente invasivas. A sua natureza mole e sem fios permite a navegação para alvos de difícil alcance do corpo humano sem danificar os tecidos circundantes. Além disso, a atuação magnética é livre de radiação, não é prejudicial para os seres humanos e elimina a necessidade de ter uma fonte de energia a bordo do robô. Apesar dos recentes desenvolvimentos no projeto e atuação deste tipo de robôs, existem alguns desafios, como localização, perceção e planeamento de caminhos, a serem superados para poderem realizar tarefas em ambientes desafiadores.

O objetivo principal do projeto é alcançar o controlo de movimento em malha fechada e o planeamento de um soft robot, o Milípede, usando imagens de ultrassom. Neste estudo, integramos estratégias de localização e controlo num sistema de atuação magnética para direcionar com segurança o soft robot para um alvo. Em relação ao controlo, um controlador Proporcional Integrativo (PI) é usado para calcular as velocidades lineares e angulares para conduzir o robô pelo espaço de trabalho evitando obstáculos. Consoante as velocidades, o campo magnético correspondente é aplicado, utilizando um conjunto com seis bobinas eletromagnéticas. A localização é obtida primeiro de uma câmara a olhar para o espaço de trabalho como prova de conceito dos métodos de controlo e planeamento de movimento. Em seguida, comparamos o desempenho entre dois algoritmos de ultrassom, um geométrico e uma abordagem de aprendizagem profunda, para estimar a pose do Milípede. Por fim, o controlo de circuito fechado do soft robot é obtido, utilizando imagens de ultrassom.

Os resultados mostram a possibilidade de usar os soft robots para realizar tarefas de forma autónoma em cenários clinicamente relevantes. A experiência de prova de conceito com a imagem da câmara de visão superior do espaço de trabalho alcançou um erro de desvio entre a posição do Milípede e o caminho planeado de 1.71 ± 1.07 mm. As abordagens geométrica e de aprendizagem aprofundada obtiveram um erro médio de estimação da posição e orientação de 0.68 ± 0.50 mm e $2.3 \pm 3.4°$, respetivamente, num conjunto de dados com 9.500 imagens de ultrassom. Por fim, o controlo em malha fechada do Milípede, usando a localização baseada em aprendizagem profunda nas imagens de ultrassom, alcançou um erro de 1.6 ± 0.84 mm e um erro da orientação de 2.23 ± 1.48 mm. Além disso, o erro de desvio médio entre a posição do robot e a trajetória planeada foi de 1.87 ± 0.92 mm.

Palavras-Chave:

Intervenções Minimamente Invasivas, Soft Robots, Atuação Magnética, Controlo de Movimento, Localização, Aprendizagem Profunda, Ultrassom

List of Figures

List of Tables

Table of Contents

Chapter 1 (Introduction)

Untethered magnetically-actuated soft robots have the potential to revolutionize minimally invasive interventions. The magnetic actuation is not harmful for humans, and its soft, untethered nature allows the navigation to enclosed and hard-to-reach targets. Furthermore, the localization of the robots in the workspace is essential for its interaction with the environment. The combination of the position and orientation of a robot is called pose, which is used in navigation and motion planning strategies.

The localization of robots can utilize several imaging approaches. For instance, camera imaging and other clinical-relevant imaging techniques, such as Fluoroscopy, Magnetic Resonance Imaging, Computed Tomography (CT) scan and Ultrasound (US). This thesis focus on the localization and closed-loop control of soft robots using camera and ultrasound imaging feedback. Ultrasound is radiation free, allows the visualization of soft materials, and has a higher acquisition rate. Even though there are already several algorithms for the localization of untethered medical robots, most of the existing methodologies lack on the estimation of its orientation. Therefore, different strategies for estimating not only the position but also, the orientation of the robot are needed for motion control and planning.

The main goal of the current project is to autonomously control a soft robot using a clinically-relevant imaging technique, the ultrasound. However, the high signal-to-noise ratio of the ultrasound image and the proximity between the densities of the robot and the medium are challenges to overcome in the estimation of the position of the robot. Therefore, to achieve the main goal several objectives have to be fulfilled:

- first, develop a camera-based position estimator to be the ground-truth of the ultrasound algorithms and demonstrate the control of the robot to follow a path generated by a path planning algorithm.
- second, build a framework capable of handling the communication between the magnetic actuation system and the ultrasound machine.
- third, assemble a convolutional neural network for the regression of the orientation and centroid of the robot in a 2D ultrasound image. Thus, construct a dataset, and train, validate, and test the architecture of the network.
- finally, investigate the performance difference between a problem-specific geometric algorithm and a convolutional neural network to estimate the pose of the robot with ultrasound feedback.

The thesis is structured as follows. Chapter 1 begins with the state of the art of untethered medical robots, tracking methodologies and path planning algorithms. Chapter 1 ends with a background over concepts discussed in this dissertation. Chapter 2 is divided into the materials, methods, and the results. First, it is provided a description of the materials used, namely the soft robot, the magnetic actuation system, and the ultrasound machine. The methods include a camera based algorithm, and two ultrasound tracking approaches which include the use of a geometric algorithm, and a convolutional neural network (CNN) to estimate the pose of the soft robot. The end of Chapter 2 presents the results of the performance of the proposed tracking strategies and the demonstration of closed-loop control using ultrasound imaging. Chapter 3 discusses the findings and shows topics of future work.

1 State of the art

In this section, it is presented several studies using untethered robots aiming medical applications. The different methodologies to estimate the pose (position and orientation) of objects in a 2D environment are enumerated, either with image processing algorithms or with convolutional neural networks. Also, the existing path planning algorithms to guide the robot to a target site, while avoiding obstacles, are described.

1.1 Surgical robots

Untethered medical robots have the potential to access more complex and smaller regions of the human body, when compared to tethered medical devices, such as flexible endoscopes and catheters, while being minimally invasive. Small-scale untethered medical robots can be classified as millirobots and microrobots depending on their size. The microrobots have all dimensions between 1 μm and 1 mm, and therefore reach submillimeter size regions in the human body [1]. Their applications include single-cell manipulation, tissue engineering, and drug delivery. However, the present study focus on millirobots where the dimensions of the robot ranges from 1 mm to less than a palm size.

Untethered pill-size capsule endoscopes with an on-board camera were first commercialized and used in hospitals in 2001 [2]. It allows the exploration of the gastrointestinal (GI) tract to previously impossible to reach regions such as the small bowel, while reducing the discomfort and sedation of the patient. However, its clinical application is bounded to passive monitoring of the GI tract since it allows optical imaging, but physicians have no control over the movement and function of the capsule. Therefore, several groups proposed improvements on the control and function of the endoscopic capsules, such as drug delivery [3], active locomotion using legs [4] or a magnetic-propeller system [5]. Despite being more portable, the on-board actuation systems limit their power source and thus the length of the procedure. Also, the on-board source of energy and the mechanical components limit the size of the capsule as well as the internal space available of the capsule. Therefore, the development of actuation mechanisms outside the capsule environment was necessary. The use of an external magnetic system for the actuation of a magnetic endoscopic capsule has the advantage of no on-board actuators, mechanisms, and batteries [6]. Previous studies show the actuation of magnetically actuated endoscopic capsules using an MRI [7] or external electromagnetic system [8, 9]. The endoscopic capsules are made of rigid structures that might damage the surrounding tissues during the magnetic actuation. Despite the improvements, the robots have limited mobility due to the inability to conform to the surroundings.

Soft robots have the advantage of being flexible structures capable of large deformations, which reduces the risk of damage when compared to rigid-bodied robots. Recent studies focus on using small-scale soft-robot for medical applications, where the robots are entirely made of soft and biocompatible materials [10, 11].

Hu *et al.* [10] developed a multi-modal small-scale magnetically actuated robot capable of walking and rolling on solid surfaces, jumping over obstacles, and swimming inside and on the surface of liquids. Additionally, it can perform cargo release and pick and place tasks. The robot consists of a rectangular sheet shape with a single wavelength harmonic magnetization. By varying the intensity and direction of the magnetic field, the robot presents different shapes, which allows the different locomotion modes.

Lu *et al.* [12] designed a millimetre sized robot with multiple legs with flexible tapered feet, which allows superior adaptability to wet and dry surfaces. Furthermore, despite the strong carrying capacity, the robot does not present a grasping function. The previous robots designs show a uniform magnetization profile and the use of an external permanent magnet, which limits the potential for control.

Venkiteswaran *et al.* [11] designed two magnetically actuated soft robots with independent locomotion and grasping functions. Additionally, it is inspired in biological motion patterns to adapt to a specific environment. The Millipede has a tail gripper, while the Hexapod has a flower gripper. The grasping function is decoupled from the locomotion by varying the direction, magnitude, and frequency of the external magnetic field. It is also shown the teleoperation of both soft robots to pick and place tasks by grasping and releasing objects. The Millipede is used throughout the current project.

1.2 Tracking

The scope of the project only includes the estimation of three position parameters. Therefore, this section presents several studies using tracking methodologies to estimate the position and the orientation of objects in a 2D image. Geometric and deep learning methodologies were addressed.

1.2.1 Geometric Methodologies

The geometric methodologies rely on imaging processing algorithms to recognize the silhouette of the object and thus, estimate the centroid and the orientation [13–15]. In the literature, the detection and estimation of the pose of untethered robots focus mainly on extracting the position of the robot, disregarding its orientation. Several authors show tracking of untethered microrobots either in Red-Green-Blue (RGB) or ultrasound images, where the orientation of the robot is considered aligned with the applied magnetic field. Therefore, extracting the pose of untethered robots requires additional estimation of the orientation parameter.

Ongaro *et al.* demonstrated pick and place tasks of a magnetically actuated soft small-scale gripper using camera feedback [13]. The tracking of the robot on the 2D environment consists on extracting the contours of the robot and then estimate its position. The first stage includes the conversion of the image from RGB to Hue-Saturation-Value (HSV) color space to achieve robustness to glare. After using a median filter to remove speckle noise, the image is binarized using a Gaussian-window threshold. Then, the contour points are mapped to the imaginary plane to calculate the Discrete Fourier Transform (DFT) coefficients which provide the position, scale, and orientation of the gripper. The contour of the gripper has a 6-fold symmetry, and a specific threshold ratio between the contour and the area of the contour. The gripper is localized with an average positioning error of 0.03 body lengths.

Vrooijink *et al.* propose segmentation techniques to track the centroid of the robotically actuated delivery sheath (RADS) on a 2D ultrasound image [15]. First, the image is convoluted with a 2D Gaussian kernel to reduce speckle and smoothen the edges. Then, a random sample consensus (RANSAC) strategy is used on the contours detected by the Canny edge detector. RANSAC fits the set of points to a parametric description of the semicircular model of the tip of the RADS. Since the fitting can be computational costly, it is conducted a study on the number of iterations of RANSAC and the centroid localization error. The results show a steady segmentation error of 0.4 mm when increasing the number of iterations.

Nguyen *et al.* use a Principal Component Algorithm (PCA) to extract the orientation and centre points of millimetre sized robots, including catheter tip, from the contour points of an X-ray image [14]. The algorithm is suitable for tracking when the robot has symmetric distribution along the major and minor axes. The distance error between the tips appearing in the X-ray image and the tip detected by the PCA is minimized to 0.2-0.4 mm, while the orientation error is 1-2°. Although, the orientation estimation is sensitive to noise and the error of the estimation increases in case of poor segmentation of the image.

A geometric solution with segmentation thresholding and contour finding method was chosen to estimate the pose of the robot in camera frames images. The simplicity and the lower noise of the images allow the use of geometric solution. Also, a similar solution was employed using ultrasound images to access the ability to estimate poses in noisier environments.

1.2.2 Deep Learning

Since the localization methodology proposed is based on convolutional neural networks, this section overviews the use of CNN for the detection and localization of objects. It is described the use of CNN to solve object detection tasks, where several objects in an image are found and classified. Also, the detection task is extended to additionally estimate the orientation of the object (pose estimation). Moreover, the state-of-the-art of the use of CNN as backbone in object detection models is presented.

In 2012, AlexNet [16] is developed as an 8-layer convolutional neural network with five convolutional layers where some are followed by max-pooling layers. The other three layers are fully connected layers with a final 1000-way softmax. Rectified Linear Units are used as activation functions to add-non-linearity to the network, instead of using sigmoid and tanh activation functions.

The network VGG-16 [17] was released in 2014, and it is a CNN with 16 layers, where 13 are convolutional layers and the remaining are fully connected layers using Rectified Linear Units (ReLU) activations. The neural network provides more layers than the AlexNet, while using smaller filters. Consequentially, it includes more parameters than the AlexNet, 138 million compared with 60 million.

One year later, GoogLeNet [18], also known as Inception, is introduced as a 22 layers deep network. The structure of the network is different from previous methods such as AlexNet and VGG-16 because it uses inception blocks. These blocks utilize multiscale convolutional transformations with filters of different sizes. Also, it is used global average pooling instead of fully connected layers. Despite the increase of the number of layers, the network keeps the same computational cost because it estimates fewer parameters, 5 million.

The increase of layers in convolutional layers does not necessarly increase the performance of the network. At some point the accuracy gets saturated, and the performance drops off. So, in 2016, He *et al.* [19] developed ResNet, a residual network to skip connections between neurons. It is used Batch Normalization (BN) after each convolution and before the activation to increase the speed of the network, while achieving the same accuracy [20]. The variants of the ResNet depend on the number of layers, for instance, the ResNet-50 has 50 layers and approximately 26 million parameters. Moreover, the memory requirements for storing the extracted features are reduced when compared with previous networks like VGG16 [17]. The feature extracted from ResNet is 2048-dimension, which is half of the traditional 4096-dimensional feature vector. Therefore, ResNet presents lower computational complexity and superior performance over previous deep networks [19].

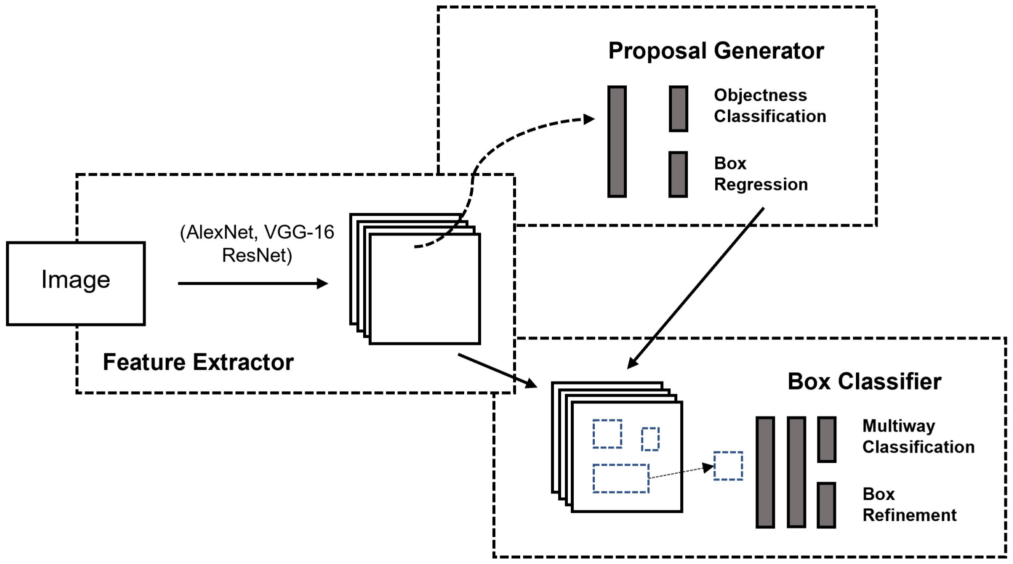
Object detection algorithms predict the bounding box and the class of each object presented in an image with multiple objects with one or multiple classes. The R-CNN (Regions with CNN features) [21] was one of the first neural networks to overcome the performance of feature extractor algorithms, such as SIFT [22] and HOG [23], for object detection. The algorithm extracts around 2000 region proposals from the input image and computes a feature vector for each proposal using a CNN. Finally, it classifies each region with a class specific linear SVMs. However, the R-CNN is computational expensive since it computes features multiple times.

The Faster R-CNN introduces a new module to the Fast R-CNN [25] to increase the efficiency and accuracy of object detection. The Faster R-CNN [26] is composed by three modules: the feature extractor using the VGG-16, the region proposal network, and the box classifier, as shown in Fig. 1a. The RPN is a fully convolutional network that uses the intermediate layer of the feature extractor to propose the class and the bounding box of the objects. Then, the box classifier uses the bounding box proposals to select the features of the intermediate layer and continues the feature extractor to predict the class and refine the bounding box. Basically, the RPN module is used to indicate the features to consider to the Fast R-CNN neural network. Several authors use a single feed-forward convolutional neural network, as pictured in Fig. 1b, such as the You Only Look Once (YOLO) [27] or Single Shot Multibox Detector (SSD) [28], to reduce the computational cost even more. Both utilize a single convolutional network to simultaneously predict multiple bounding boxes and the respective class probability. However, the SSD model adds several convolutional feature layers to the end of a base network. The layers decrease in size along the network, which allows predictions of object detections at different scales.

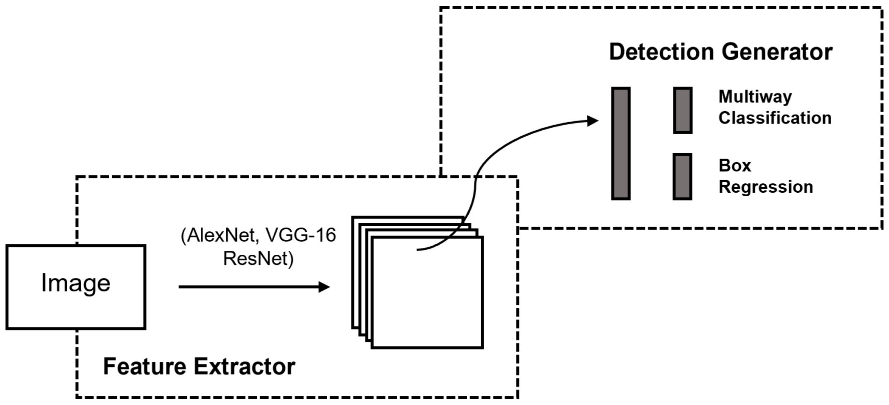
Regarding object localization, the position as well as the orientation (pose) of the object are required. The use of CNN for pose estimation is mostly based on previous works about object detection.

Wang *et al.* utilize a single shot detector architecture to simultaneously detect ships and estimate the angle in Synthetic Aperture Radar (SAR) images [29]. The angle estimation problem is a regression task integrated into the bounding box regression from previous studies [28]. The bounding box regression uses a translation of the center point of the bounding box and a transform of the height and width in log space. The added angle regression term utilizes the tangent to calculate the offset between the estimated and labelled angle. The regression module uses multi-oriented prior boxes with different sizes, ratios, and angles to make the model learn to estimate the pose of the ships. The results show that integration of the angle estimation directly into the object detection pipeline provides a higher accuracy and lower computational time when compared to two-stage methods.

Liu *et al.* propose a convolutional neural network to estimate head poses using trained synthetic head



(a) Faster R-CNN



(b) SSD

Figure 1: Comparison between two CNN object detection architectures. Both architectures use a feature extractor. However, the Faster R-CNN utilizes a proposal generator to predict bounding boxes and class proposals that are classified by a box classifier, while the SSD performs both steps in a single detection network. The figure is adapted from [24].

images [30]. The head pose estimation is considered as a regression problem of the three angles: yaw, pitch, and roll. The loss function is the Euclidean distance between the labelled head pose, and the predicted head pose by the CNN proposed.

Hara *et al.* propose a CNN, with ResNet-102 as backbone, to estimate the orientation of objects in an image [31]. The orientation estimation is converted into a set of separated discrete orientation estimation tasks, where each one uses a standard softmax function. Then, the discrete orientation outputs are converted to a continuous orientation prediction using a mean-shift algorithm.

Baltruschat *et al.* propose a neural network to estimate the orientation of the anatomy in a digital X-ray image [32]. The backbone is a ResNet-50 to extract features for the angle regression. The angle loss function handles circular angle representations by representing the angles in the complex plane. The loss function is an empirical average between the label and the angle prediction in the complex plane using $z(\theta) = (cos(\theta), sin(\theta))$. The performance is increased when the layers of the ResNet-50 are retrained rather than using the weights from the training on the ImageNet dataset [33].

There are several studies that compare the performance of the convolutional neural networks used as backbone in object detection. In 2016, Lee *et al.* compare the performance of R-CNN based on the input image size, detection precision, and computational time [34]. It is clear that time-consuming models with bigger image input size tend to achieve higher precision. The precision of the models used in the object detection comparison is ResNet > VGG16 > GoogLeNet, but the computation times are reversed of the previous order. More recently, Amjoud *et al.* compared the performance of different object detection models using different networks as backbones [35]. The results showed a higher performance of VGG-16 and the ResNets family networks when in comparison with the AlexNet and GoogLeNet networks.

A state-of-the-art convolutional network was used as backbone of the network proposed and modified to classification and/or regression problems. Each branch of the regression added has a loss function which is responsible for converging the predicted value to the label. Therefore, we chose to build a CNN, using a ResNet-50 as backbone and a regression branch for each of the pose values. The ResNet-50 network was used because it presents a higher accuracy and estimates a lower number of parameters.

1.3 Path Planning

The application of untethered milli/micro robots to clinical procedures requires an obstacle-free path to autonomously guide the robot to a target. The choice of the suitable path planning algorithm depends on the computational cost and path distance and time available for each application [36].

Scheggi *et al.* performed a comparison between six path planners in a 2D dynamic environment using a paramagnetic microparticle and comparing deterministic and probabilistic path generators [37]. The deterministic path planning, such as A* with uniform grid and quadtree, D* Lite and Artificial Potential Field (APF), have a simpler implementation.

The A* algorithm [38] is an improved version of the Dijkstra algorithm [39] by adding a heuristic cost, which is the distance between current and destination nodes. The Dijkstra algorithm is a method to find the shortest path between two nodes in a graph. Therefore, the A* algorithm attributes a cost to each node of the grid map, which is the sum of Dijkstra and heuristic costs, and searches a path through the minimization of a cost function. The cost function can be modified to fit the needs of the applications, such as increasing the distance between the obstacles and the path generated. By modifying the cost function of each node, Lim *et al.* showed a performance increase in the driving time and distance during the autonomous control of a microrobot [40]. The path searching algorithm generates fast and accurate paths at low spatial dimensions, while computing the shortest possible path from the starting node to the target node. However, for larger maps, the algorithm requires a lot of memory, since it needs to store multiple states for each node [37].

The D* Lite algorithm [41] was developed to increase the performance of the A* algorithm by reusing the information from previous searches. Initially, the D* Lite runs similar to the A* but as the robot starts to move to the target, the algorithm recalculates the best path to the target faster than the A*. Lifelong Planning A* (LPA*) [42] is used to recalculate the path by updating the costs of the nodes from the previous search.

The Artificial Potential Field [43] method consists of an attractive potential field towards the target and a repulsive potential field generated by the obstacles in the map. The sum of the two potentials results in the potential of the robot, and the robot follows the negative gradient of the resulting potential. Despite not being computationally expensive, the APF method can trap the robot in a local minimum of the potential field, and it is not guaranteed to find the optimal path.

On the other hand, the deterministic approaches such as Probabilistic Roadmap (PRM) and Rapidly-exploring Random Trees (RRT) provide solutions to vast, high dimensional configuration spaces, but the execution time tends to infinite, if no solution exists.

The Probabilistic Roadmap [44] planner is processed in two phases: a leaning phase and a query phase. During the learning phase, a PRM is constructed, which consists of a graph with nodes that correspond to collision-free configuration and with edges corresponding to feasible paths. In the second phase, the start and target configurations are connected to the graph of the roadmap and the shortest path between the two nodes is computed.

The Rapidly-exploring Random Trees [45] algorithm grows a tree rooted at the initial state of the robot. At each iteration, the method selects a random state from the free space of the map, finds the closest node in the tree, and computes a feasible control input to minimize the distance between the random state and the nearest state. The new state is obtained by applying the input to the dynamic model of the robot and adding it to the tree. The algorithm stores the new state as well as the corresponding input. Therefore, the RRT finds a feasible path because it computes the inputs necessary to reach the target, according to the dynamics of the robot.

The results show that there is no significant statistical difference between deterministic and probabilistic planners due to the low dimensional configuration space of the environment and the simple dynamic model of the microagent. Also, the A* with quadtree and APF achieved the lowest computation time [37]. In another study [46], the deterministic path planning algorithms outperform the probabilistic in 2D environments. The performance of the different algorithms was evaluated based on the length and smoothness of the path, the computational time, and the success rate of the planning. However, the results were obtained in simulations and did not consider the dynamics of the robot to move.

Moreover, the path planning algorithms can be extended to 3D non-static environments, such as steerable needles guidance to a biopsy target. A RRT motion planner uses the needle tip position information to compute and update a feasible path to the target, while avoiding static and moving obstacles [47].

In this project, the path planner choice was the A*-WAPP, because of the fast and accurate path generation at low spatial dimensions, while computing the shortest possible path. Also, the complexity of the workspace did not require the use of probabilistic path planners.

2 Background

The two methodologies for the detection of the soft robot are based on geometric algorithms and deep learning methods. Therefore, it is important to address concepts used in both strategies. First, it is introduced the basis of geometric algorithms, such as image processing and calibration processes. Then, it is described the fundamental principles of convolutional neural networks.

2.1 Geometric Principles

2.1.1 Image Processing

Images are composed by pixels, where each pixel is composed by a value in the colour space representation. Usually, each pixel has an 8-bit resolution and the values range from 0 to 255. The most common colour representations are in greyscale, RGB and HSV. In greyscale, each pixel is a single channel which represents the amount of light. The intensity of the pixels ranges from black, the lowest value, to white, the higher value. The RGB colour space is a three component representation which matches the perception of the human eye. The human eye contains three pigments, each sensible to either red, green or blue. A similar process happens in the RGB space, where the addition of the value of each component leads to a specific colour. The HSV colour model is an alternative cylinder representation. The Hue is arranged in a radial slice measured in $[0°, 360°]$ which corresponds to the colour. The Saturation is responsible for the "purity" of the colour. The lower saturation of a specific hue corresponds to the pure gray on the greyscale. The Value is the height of the cylinder, which represent the brightness of a colour. The HSV colour space is more robust to brightness changes than the RGB colour, and it is preferably for colour threshold [48].

In computer vision, image filtering changes the range of the pixel values, so that the colours are altered without changing the position of the pixels. The goal of filtering is to modify the image properties to extract information, such as edges, corners, and blobs. Two commonly implemented filters are the image segmentation and image convolution. Image segmentation is used to separate an image into regions with similar pixel values. The segmentation is useful for an easier identification of objects and their boundaries. The threshold of an image is an example of image segmentation, which consists of choosing upper and lower limits of colour components in order to obtain a binary image. If the pixel value is within the range of the limits, it receives the value one, otherwise it receives no value. The threshold is performed over the number of channels of the image.

On the other hand, the convolution between an input image and the kernel filter is used for blurring, sharpening, and edge detection operations. The inverted kernel slides over the input image and performs an element wise multiplication between the overlapping pixels and then, sums up the results into a single output pixel. The values of the kernel matrix define the changes applied to the input image. A median filter selects the median value from each pixel's neighbourhood, depending on the size of the kernel. A median filter is useful for removing salt and pepper noise. Another type of kernel is smoothing filters, which are used to remove high frequencies. Examples include mean filters and Gaussian filters, which performs a weighted mean filter according to the values of a Gaussian function. The gradient of the image intensity function is computed using a Sobel filter. The Sobel operator is a separable combination of a horizontal central derivative and a vertical smooth filter. The Sobel filter with horizontal derivative detects horizontal edges, while the vertical detects vertical edges [48]. The Canny edge detector uses a multistage algorithm to detect edges in an image [49]. It starts by smoothing the image using a Gaussian filter. Then, the intensity gradients of the image are calculated using the Sobel filter in both horizontal and vertical directions. The edge magnitude and direction of each pixel are calculated, using both gradient images. The unwanted pixels are removed using a non-maximum suppression of the magnitude values over the direction of the gradient. The final binary image presents the edges of the objects detected.

2.1.2 Calibration

The calibration process consists of finding the relationship between image coordinates and world coordinates of coplanar points. The homography matrix is a projective transformation which is estimated using two sets of points corresponding to both projective coordinate frames. Beginning with the homography between world points (x_i^w, y_i^w) with known coordinates and the respective image points (x_i^c, y_i^c), eq. 1 is obtained. The equation is expanded and the parameters are rearranged to get eq. 2, which results in two equations and nine unknown variables of the homography that were obtained from a single point correspondence.

$$\lambda \begin{bmatrix} x_i^w \\ y_i^w \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x_i^c \\ y_i^c \\ 1 \end{bmatrix} \tag{1}$$

$$\begin{bmatrix} x_i^c & y_i^c & 1 & 0 & 0 & 0 & -x_i^c x_i^w & y_i^c x_i^w & -x_i^w \\ 0 & 0 & 0 & x_i^c & y_i^c & 1 & -x_i^c y_i^w & y_i^c y_i^w & -y_i^w \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \\ h_{33} \end{bmatrix} = 0 \tag{2}$$

At least 4 + 1/2 points are necessary to determine the parameters of the homography. Therefore, more points are used and, consequently more equations are stacked in the left matrix of eq. 2. The set of equations of the form $Ah = 0$ is overdetermined and can be solved using singular value decomposition of the matrix A.

$$A = U\Sigma V = \sum_{i=1}^{9} \sigma_i u_i v_i^T \tag{3}$$

Ideally, if the matched points did not contain any error, the solution would be the null singular value of the right singular vector v_i. However, the matching of points is not perfect and the solution is the smallest value in the singular vector. Moreover, the estimation of the homography is improved using RANdam SAmple Conscensus (RANSAC) [50]. The RANSAC loop randomly selects four correspondences and computes the homography. Then, it uses the homography to map the points of the image to the world and counts the number of inliers, when comparing to the correspondent points. The homography with the largest number of inliers is kept.

2.2 Deep learning

2.2.1 Artificial Neural Networks

Artificial neural networks are a biologically inspired programming algorithm which enables a computer to learn based on observational data. The network presents multiple layers of neurons, which are connected by synapses. Fig. 2 shows three types of layers. The input layer is the set of input sites that gather information from the outside. The output layer is the set of output units responsible for presenting the desired output. The hidden layers are all the layers that have no direct connection with the outside and process the information to predict results. A fully connected layer is when all neurons from a layer are connected to each neuron of the next layer. Also, the neurons of a network are not usually connected to each other [51].

Regarding the communication between layers, the synapse weights the information output of the neurons from the previous layer (w_i, x_i) and transfer it to the neurons of the next layer [52]. Each neuron collects the information received and outputs a value based on an activation function f and a bias b, as shown in Fig. 3.

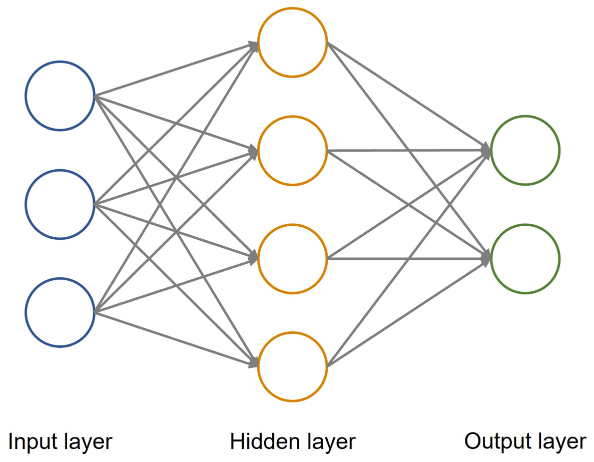


Figure 2: Concept of the architecture of a neural network. The input layer gathers the information from the data, it is processed through the hidden layer and the output layer predicts the class of the data.

The forward propagation for a layer with n inputs and m outputs is described as a matrix multiplication, $y = f(W.x + b)$, where $x \in \mathbb{R}^n$ represents the outputs of the previous layer, $W \in \mathbb{R}^{m \times n}$ the weight matrix, $b \in \mathbb{R}^m$ the biases of the neurons and $y \in \mathbb{R}^m$ the output of the layer [52]. The ReLU and the sigmoid

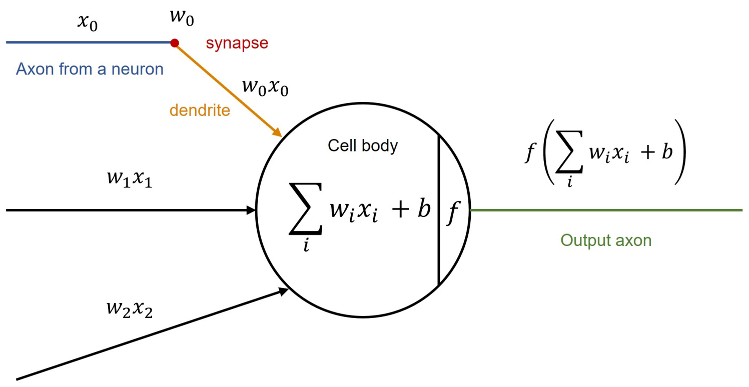


Figure 3: Mathematic model of a neuron. The information from previous neurons x_i is weighted by synapses w_i. The output of the neuron is defined by the sum of the weighted inputs with a bias b and an activation function f.

functions are the most used activation functions for deep neural networks.

$$f_{ReLU}(x) = \max(0, x) \tag{4}$$

$$f_{sigm}(x) = \frac{e^x}{1 + e^x} \tag{5}$$

The ReLU function is similar to a linear unit, but the output of the function is zero across half of the domain. On the other hand, the sigmoid function is very sensitive to input values near to zero and it saturates across most of the domain. It saturates to a higher value when the value is very positive and to a lower value when the value is very negative [52].

During training, in each prediction, the input data is processed through the network using the current weights and biases to generate an output. The output is compared with the ground truth using a loss-function specific to the problem to solve. Afterwards, a backpropagation algorithm computes the gradient of the loss function one layer at the time [53]. Then, optimizers use the gradients to change the weights of the network to minimize the value of the loss function.

The dataset is usually divided into training data and validation data. The validation dataset is used to verify if the neural network can transfer the knowledge learnt in the training set to new data. In classification problems, the performance of a neural network on the validation dataset is evaluated using the precision and accuracy:

$$Precision = \frac{TP}{TP + FP} \tag{6}$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{7}$$

where TP is the number of true positives, FP false positives, TN true negatives, and FN the false negatives.

A good performance of the training of the network is achieved if the validation loss is inferior to the training loss. However, bad training may occur, leading to underfitting or overfitting. Underfitting occurs when the model can not get a sufficiently low error on the training set. On the other hand, overfitting occurs when there is a big difference between the training and validation error, the training accuracy improves but the validation accuracy drops. Moreover, the network learns features that are specific to the training set without learning the general concepts to classify the data.

2.2.2 Convolutional Neural Networks

Convolutional neural network (CNN) is basically a neural network that uses convolution instead of matrix multiplication in at least one of the layers [52]. Mathematically, the convolution between the image, $I \in \mathbb{R}^{i \times j}$, and the kernel, $K \in \mathbb{R}^{m \times n}$, is given by

$$F(i, j) = (K * I)(i, j) = \sum_{m} \sum_{n} I(i - m, j - n) \cdot K(m, n) \tag{8}$$

resulting in a feature map, $F \in \mathbb{R}^{i \times j}$.

The CNN uses the convolution as a feature extractor, by sliding a kernel over the image to obtain a feature map of the image. Each convolutional layer is composed by multiple kernels, which generate several feature maps, as presented in Fig. 4. During the training, the CNN learns the weights of the kernel.

A convolutional layer performs the convolution between each input channel, $k \in [0, c_{in})$, of the input feature maps, $X \in \mathbb{R}^{c_{in} \times h_{in} \times w_{in}}$, and the corresponding kernels of the weights learnt, $W \in \mathbb{R}^{c_{in} \times c_{out} \times h_{kernel} \times w_{kernel}}$. The variables c, h, and w correspond to the number of feature maps, the height, and the width, respectively. The process is repeated for each output channel, $l \in [0, c_{out})$, producing the output feature maps, $Y \in \mathbb{R}^{c_{out} \times h_{out} \times w_{out}}$, of the convolutional layer. Also, a bias $b \in \mathbb{R}^{c_{out}}$ is added to the output feature maps, and it is applied an activation function f [52, 54],

$$Y(l) = f(b(l) + \sum_{c_{in}-1}^{k=0} W(l, k) * X(k)) \tag{9}$$

Another layer present in convolutional neural networks is the pooling layer, which reduces the size of the feature maps with a statistic summary of the neighbouring pixels. Thus, leading to a decrease of the parameters and computation in the network. For instance, the most common is the max pooling layer, which outputs the maximum value within a rectangular neighbourhood. The number of feature maps increases for deeper layers to allow the learning of more concepts. The convolutional layers learn hierarchical information, where the first layers recognize simple features, such as edges, and the deeper layers learn smaller details. For classification problems, the CNN usually ends with a fully connected layer with as many outputs as classes to be predicted, as shown in Fig. 4.

Moreover, CNNs can solve problems using classification or regression. A classification prediction model is the task of estimating discrete output variables, i.e. classes. The classification model predicts a probability for each output class. Then, the predicted class with the higher probability is converted to the estimated class value. Examples of a classification problem include estimating the class of objects presented in an image. The classification accuracy is the percentage of correctly predicted classes out of all the predictions made. On the other hand, a regression prediction model predicts a continuous output variable within a given interval of values. For instance, the prediction labels can be position and orientation values. Since the regression model predicts a quantity, the accuracy is measured with the error between the predicted and the labelled values. The root mean squared error (RMSE) is one of the most common functions to calculate the error, since the error is in the same units as the predicted value.

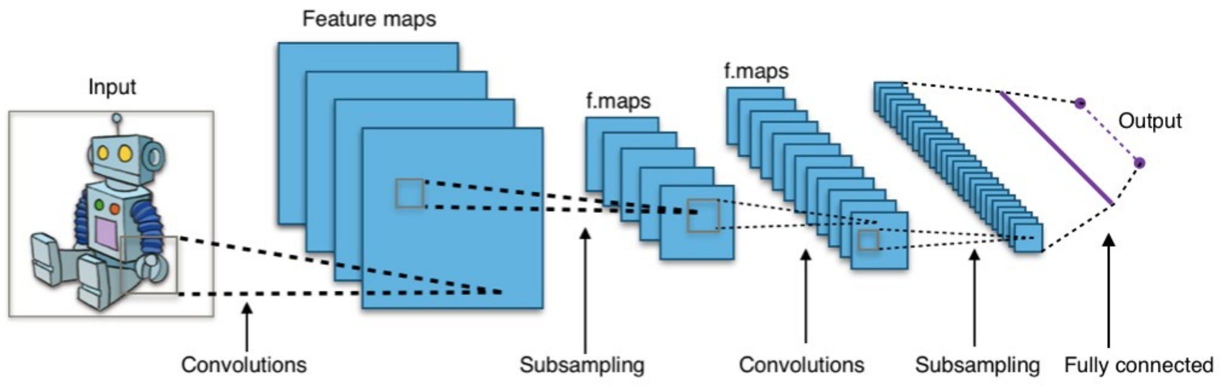


Figure 4: Typical architecture of a convolutional neural network (CNN). The convolutional layer extracts features maps, which are downsampled using a pooling layer. The fully connected layer outputs the prediction of the network. The image was obtained from [55].

Chapter 2 (Materials, Methods and Results)

1 Materials

1.1 Magnetic Actuation System

BigMag is a magnetic actuation system with a moving array of six electromagnetic coils on two symmetric mobile frames. The two frames rotate around a spherical workspace with a diameter of 10 cm [56]. Using an inverse field map technique, the system maps the required magnetic field to BigMag configuration variables, namely, the currents to generate in each coil, and the angles of the bottom and top frames holding the coils. The inverse map was validated and showed a mean magnetic field error of 2.20%. Moreover, it generates any desired magnetic field up to 60 mT with a bandwidth of 40 Hz within the workspace [57]. Also, BigMag is equipped with 2 Dalsa Genie Nano-C1940 (Waterloo, Ontario, Canada) cameras, which provide a top and a side view of the entire workspace [56].

The control system of BigMag is implemented in C++ on a computer running Linux Ubuntu 14.04.01, equipped with Intel Xeon E5 CPU, NVidia Quadro K4200 GPU, and 32 GB RAM. The control cycle is set to 10 Hz to eradicate any jitter generated by the variable acquisition time of the camera frames (up to 60 ms) [57]. In this project, we used the BigMag system to generate the intended magnetic field. The control of the coils and the currents necessary to actuate the magnetic field were already implemented. Also, the connection between the BigMag system and the cameras was already implemented.

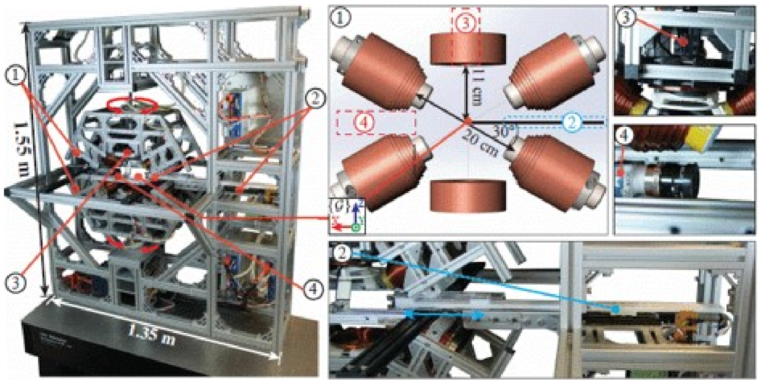


Figure 5: BigMag embodies an array of six electromagnetic coils on two symmetric rotatable frames. Left: the overview of BigMag. Right: elements of the system: ① mobile electromgnetic coils, ② inserter of continuum manipulators, ③ top camera and ④ side camera. The image is obtained from [56].

1.2 Soft Robot

In this project, the soft robots are actuated using magnetic fields. The fabrication process controls the magnetic dipoles in the robot, allowing the production of functional magnetic elements, such as the legs for locomotion. The actuation of a magnetic field on the magnetic dipole moments inside the soft robot generate magnetic torques that deform the soft material. Therefore, the locomotion gait patterns are controlled by the amplitude and direction of the actuating magnetic field [11].

The soft robots achieve locomotion by varying the magnitude, frequency, and orientation of the external magnetic field. Furthermore, the navigation of the robots is achieved by generating different displacements on either side of the robot body. In the scope of this project, only the locomotion and navigation are investigated, despite their ability to execute grasping functions. Also, the Millipede is the bio-inspired robot used for the demonstrations [11].

The Millipede robot is inspired by myriapods such as millipedes and centipedes. In nature, these animals utilize a Central Pattern Generator (CPGs) for locomotion by synchronizing limb function in groups [58]. The multiple legs of the robot achieve locomotion in a sequential wave-like pattern. A set of legs is capable of generating locomotion through an uneven terrain when it is used a metachronal rhythm [59]. In this work, the Millipede robot can steer due to the two set of legs and a longitudinal symmetry tilt in the magnetic dipoles.

The fabrication process of the Millipede is described in detail by [11] and illustrated in Fig. 6. All the magnetic components are made from a magnetic polymer composite (MPC). It is composed by a mixture of a silicone rubber matrix and a ferromagnetic powder. The MPC is subjected to a 1 T magnetic field to set the orientation of the magnetic dipoles to the desired direction. The Millipede is assembled with a top layer of silicone rubber and two set of legs individually magnetized. The magnetization of the legs is constrained with a custom-helical fixtures to create a sinusoidal magnetic profile. In this work, the robots are navigated in a water medium, which requires some modifications to increase the weight of the robot. The density of the silicone rubber is inferior to the water and leads to the floating of the robots. Therefore, a component of silicone rubber mixed with aluminium powder is used to sink the robot and generate enough friction for the locomotion of the robots.

The kinematics of the soft robots is described in Fig. 7, explaining the influence of the magnetic field on the locomotion of the Millipede. The Millipede presents two sets of magnetized legs connected by a silicone rubber body. The dipoles of each set of legs are aligned in a 30º plane angle from the vertical plane and follow a sinusoidal profile, allowing locomotion under rotating magnetic field [59]. The Millipede concept is presented in Fig. 7, and shows the sinusoidal pattern of the magnetic legs, noticing the wave contact of the legs with the substrate during the gait cycle. The symmetry of the magnetic dipoles between both set of legs about the vertical plane allows a straight-line motion, when in presence of a rotating magnetic field about this plane. Therefore, the tilting of the plane of rotation of the magnetic field by an angle (ψ) results in a different displacement on the two sets of legs due to the different magnetic torques experienced. The turning motion is achieved by controlling the displacement of each set of legs using the tilt angle (ϕ). The increase of the magnitude of the magnetic field leads to a greater displacement, increasing the speed of the Millipede.

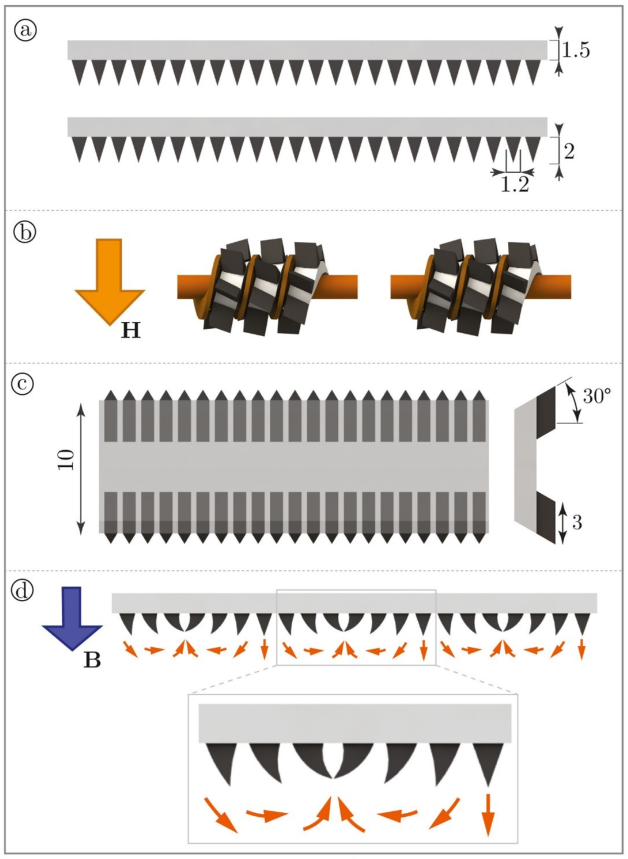


Figure 6: The fabrication process of the soft robots combines silicone rubber (white) and magnetic polymer composite (black). ⓐ The two legs of the Millipede robot are fabricated separately, and ⓑ a magnetization field (**H**) is applied while placing the helical fixtures to obtain a sinusoidal magnetization profile. ⓒ The two sets of legs are assembled to form the Millipede. ⓓ Under an external magnetic field (**B**), the magnetic dipoles of the legs (orange arrows) align with the magnetic field, causing the deformation of the legs. Dimensions are marked in [mm]. The figure is obtained from [11].

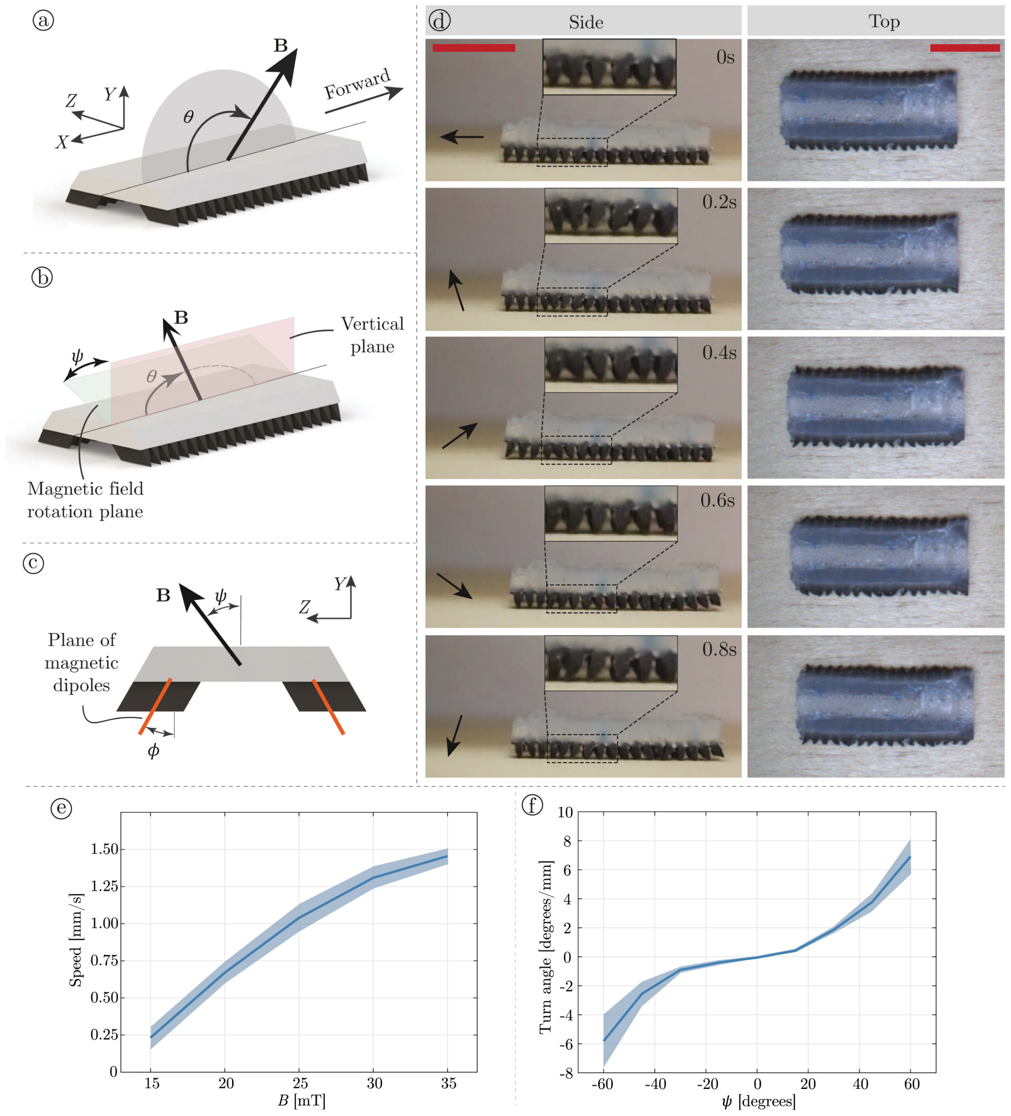


Figure 7: Kinematics of the Millipede robot by actuating a rotating magnetic field. ⓐ The straight-line locomotion consists of generating a rotating ($\theta \in [0°, 360°]$) magnetic field (**B**) of constant magnitude in the vertical plane (**XY**). For turning locomotion, the magnetic field is rotated in a plane tilted by an angle (ψ) from the vertical. ⓒ Back view of the robot showing the direction of the magnetic dipoles in the legs (aligned at angle $\phi = 30°$ to the vertical). ⓓ Images of the robot during a straight-line locomotion with the respective direction of the magnetic field (25 mT). Scale bar is 10 mm. ⓔ Speed of the Millipede robot as function of the amplitude of the magnetic field actuated. ⓕ Angular change of the Millipede robot as a function of the tilt angle (ψ) of the magnetic field. The shaded region represents the standard deviation of the mean values of three specimens. The figure is obtained from [11].

1.3 Ultrasound machine

One of the main contributions of this project was the integration of an ultrasound probe into the magnetic actuation system, BigMag. Bellow, we describe the characteristics of the ultrasound probe, the modifications performed on the Clarius Cast API to grab and process frames in real-time, and the integration of the probe into the closed-loop control in the BigMag system.

The Clarius Scanner L15 HD is a linear ultrasound machine, which transmits HD images via wireless to the Clarius App installed in a smartphone. The field of view of the scanner is 5 cm and the depth ranges from 1 cm to 7 cm, according to the operator preference. Furthermore, the Clarius App enables the user to change the ultrasound imaging mode, the application, and make annotations on the image. Depending on the depth of the ultrasound image, the transmission rate from the probe to the Clarius App varies from 25 Hz to 16 Hz. The scan time is approximately 60 min, but the recommended exam time is 20 min due to overheating of the scanner.

Also, the Clarius team provides a Cast API built in C++ with Qt Creator [60], which allows the use of the same features of the mobile application in a computer running Linux Ubuntu 20.04 and later. The communication between the probe, mobile device, and computer is shown in Fig 8. First, the Clarius App creates a primary TCP connection with the ultrasound probe. Then, the Cast API establishes a secondary connection directly with the scanner, enabling it to receive the images. The communication protocol allows the live scanning of the ultrasound images simultaneously on the smartphone and the computer. However, the Cast API has some limitations: the Cast API must be executed while the Clarius App is running and connected to the probe; all three devices must be connected to the same wireless network; and raw data can only be captured while the imaging is frozen. However, the ultrasound imaging of the soft robots requires real-time acquisition and processing of images from the Clarius probe. Therefore, we modified the cast API to allow real-time localization of the soft robot. A thread was added to the project and executed in parallel to process the image without delaying the capture of new images. The image processing consists of converting the QImage captured by the Cast API to an OpenCv image [61, 62]. Then, the OpenCv image is suitable for image processing methods, such as, segmentation and filtering. The modifications employed permitted the real-time acquisition and processing of the ultrasound images in the C++ Cast API on the laptop.

Afterwards, the Cast API was connected to the BigMag, via an Ethernet cable, and we created a UDP communication protocol to transfer information between the two systems [63]. The data is transmitted by using a buffer of characters, where the first part corresponds to the protocol situation, and the others correspond to the x-component, y-component and orientation of the robot. The protocol covers two situations described in section 2, namely, the construction of the dataSet for the deep learning approach, and the closed-loop control system using ultrasound images. The new Cast API empowers the system to process and save ultrasound images in real-time. During the image processing, the ultrasound tracking methods were used to estimate the position and orientation of the Millipede.

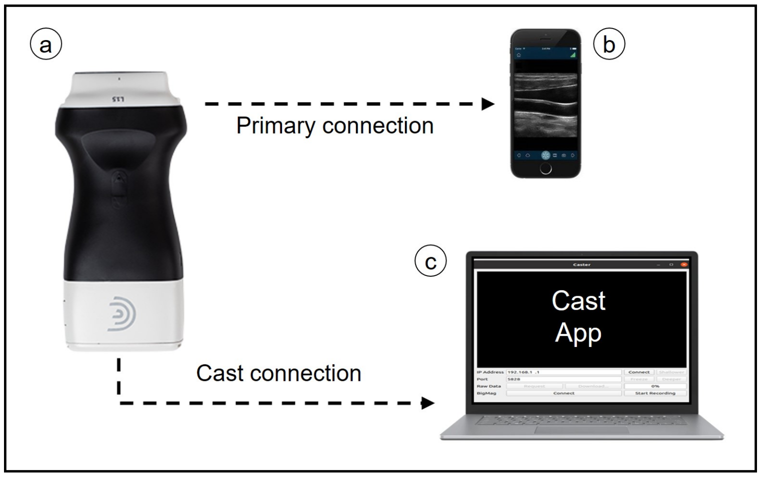


Figure 8: Communication between the Clarius probe ⓐ, the Clarius App installed on a smartphone ⓑ and the Clarius Cast API running in a computer ⓒ. The Clarius App establishes a primary TCP connection with the Clarius probe. The cast API establishes a secondary connection directly with the scanner, enabling it to receive the images.

2 Methods

2.1 Motion planning and control

The estimated pose of the robot is provided to a path planning algorithm to compute a collision-free trajectory to a target. Among the state-of-the-art motion planners, a A*- WAPP-waypoints is used. The A*-WAPP-waypoints computes points along the trajectory that are fed to the control algorithm. Then, the electromagnetic field necessary to drive the robot to each waypoint is calculated, as well as the necessary currents to apply the required field.

2.1.1 Path Planning

The A*- WAPP-waypoints is a path generation algorithm based on the A* algorithm by [38]. The modified A* algorithm reduces the possibility of collision with the obstacles, while improving the driving time distance and stability [40].

The A* algorithm utilizes the information of the heuristic cost function expressed by

$$f(n) = g(n) + h(n) \tag{10}$$

where n is the number of the node, and the fitness cost ($f(n)$) is the sum of the goal cost ($g(n)$) and the heuristic cost ($h(n)$). The goal cost is the cost required to move from the start node to the n-node, while the heuristic cost of the node is the Euclidean distance from the n-node to the goal node. The A* path planner begins the search in the start node and consists of finding the node with the smallest fitness cost among the eight adjacent nodes of the current searching node. The algorithm creates the path when the searching node is the goal node. The path generated by the A* algorithm consists of the nodes with the lowest cost value.

However, since the minimization cost function does not account for the distance to the obstacles, the path generated is too close to the obstacles, which might cause collisions. To solve it, Wall Avoidance Path Planning (WAPP) is introduced to penalize nodes near the obstacles. The A*-WAPP described by equation

$$f(n) = g(n) + h(n) + d(n) \tag{11}$$

consists of adding the distance cost ($d(n)$) to the previous fitness cost equation, eq. 10. The distance cost is inversely proportional to the distance between the node and the obstacle. The values around the obstacles are determined to allow the turning of the robot. So, the nodes adjacent to the obstacles have a higher distance cost value.

The 2D environment of the robot is divided into grids and the obstacles are placed by the user to obtain a map of the obstacles. The grid map consists of the binarization into occupied or free cell. The camera and ultrasound images are divided into grids of 30x48 and 32x18, respectively.

The tracking algorithm estimates the starting position of the robot and the user selects the goal point in the 2D environment. Then the A* - WAPP calculates the optimal path so that the robot avoids the obstacles and reaches the target.

2.1.2 Motion control

The path following consists of driving the robot to each waypoint by applying the required electromagnetic field. The position of the robot ($x_{robot}, y_{robot}, \alpha_{robot}$) and each waypoint coordinates (x_{path}, y_{path}) are used to calculate the necessary electromagnetic field to control the robot to the target. Therefore, the robot keeps a distance $d(k)$ to follow each point generated by the path planner algorithm and the error is defined by

$$e(k) = \sqrt{(x_{path} - x_{robot})^2 + (y_{path} - y_{robot})^2} - d(k) \tag{12}$$

which is the Euclidean distance between the current position of the robot and the waypoint of the path, subtracted by a distance d(k). The error is minimized to zero according to the control of the velocity of the robot using a proportional integral (PI) controller,

$$v(k) = k_v e(k) + k_i \int e(k)dt \tag{13}$$

The integral component ensures a finite velocity when the error tends to zero. The second controller turns the robot towards the waypoint, which is at an angle,

$$\beta(k) = \tan^{-1} \frac{y_{path} - y_{robot}}{x_{path} - x_{robot}} \tag{14}$$

relatively to the robot and uses a proportional controller described by

$$w(k) = k_s \left(\beta(k) \perp \alpha(k)\right), \; k_s > 0 \tag{15}$$

Therefore, the angular velocity is the angle that the robot needs to turn to the waypoint multiplied by a constant gain, k_s. The angle to turn is the difference between the angle of the robot and the angle between the position of the robot and the waypoint.

The magnitudes of the linear and angular velocities are controlled by the PI action and the performance is defined by the gains k_v, k_i, k_s. The PI controller is reset when the value of the error is below 1 mm for the waypoints of the path. The distance d is set to half the length of the robot to allow the turning of the robot and increase the steering stability. However, when the robot is reaching the target node, the d is set to 1 mm, so that the robot accurately reaches the target.

The kinematics of the soft robot described in Fig. 7 allows the estimation of the electromagnetic field, both amplitude and direction, necessary to drive the robot to each node. The commands of driving forward, backwards, and turning either clockwise or counterclockwise are defined by the error of the linear and angular velocities. So, if the error of the linear velocity is bigger than the error of the angular velocity, it is applied either the forward or backward command. Then, the magnetic actuation system generates the required currents corresponding to the required magnetic field.

2.2 Camera-based tracking

The workflow between the BigMag system and the camera-based tracking is presented in Fig. 9. The closed-loop system relies on the camera-based tracker to estimate the pose of the robot. Then, the control system uses the pose of the robot to output the magnetic field needed to move the robot along the path

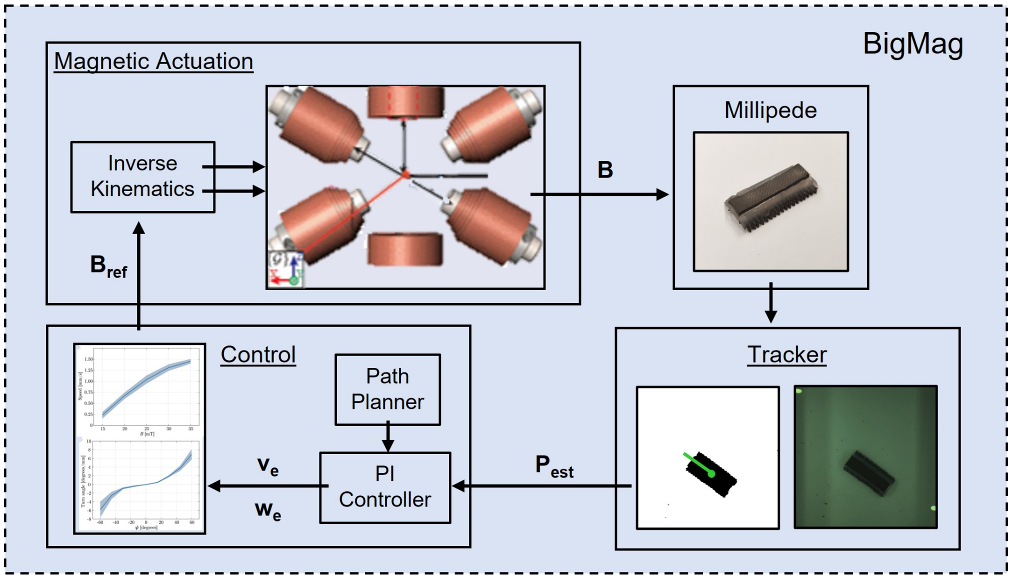


Figure 9: Block diagram summarizing the closed-loop system for controlling magnetic soft robots using camera imaging. The camera-based tracker estimates the pose of the soft robot (P_{est}), which is fed into the Proportional Integral (PI) controller. According to the linear and angular velocities errors, v_e and w_e, respectively, the control block outputs the estimates magnetic field (B_{ref}) to drive the soft robot along the path generated. Using inverse kinematics, the magnetic actuation system of BigMag generates the actual magnetic field (B) on the Millipede.

planned. The proposed robot localization is based on a fixed RGB camera, Dalsa Genie Nano-C1940 (Waterloo, Ontario, Canada) camera, looking at the BigMag workspace with a top view. The connection between the BigMag system and the cameras was already implemented. Therefore, we incorporated the tracking algorithm into the camera frame grabber to perform the processing of the camera images. On the camera frames, the image is processed into two stages which consists of a robot detection within the frame, and a position and orientation estimation, as shown in Fig. 10.

The first stage performs the detection of the contours of the soft robot from the binary image (black and white). The input image is a Red-Green-Blue (RGB) colour image acquired by BigMag at a 50Hz rate, and it is converted to the Hue-Saturation-Value (HSV) colour space. Then, each channel values are bounded using a user-friendly interface to detect the colour of the robot. The image is binarized using a Gaussian-window with the thresholds defined by the user. The values are chosen empirically until the colour space of the robot is differentiated from the background. Then, morphological opening, erosion followed by dilatation, is used to remove noise and small blobs, while keeping the shape of the robot. Afterwards, the points of the contours of the blobs in the image are obtained using a finding contour algorithm [64]. The contour of the robot is selected among the others, according to previous information, such as shape, area, and size.

The second stage consists of processing the soft robot contour points to estimate the position and the orientation. The centroid is calculated through the moments of the contours using the Green's Theorem [65] by integrating over the perimeter of the contour. Regarding the orientation estimation, the contours points are fitted to an ellipse using an algorithm proposed from [66], consisting of the minimization of an objective algebraic distance function. The angle is obtained from the major axis of the ellipse.

The position estimated is regarding the coordinate frame of the image and, therefore, needs to be mapped to the BigMag coordinate frame. The soft robot can move in a 2D environment and coincides with the origin of the BigMag coordinate frame, the z-component is null. A calibration step is required to map the coordinates from the image to the world, using a calibration board. The technique performed by [67] requires the user to interactively click the corners of the calibration board in the camera image. The matching pairs of coordinate points from both reference frames are used to estimate the homography matrix transformation from the image to the plane of the BigMag with $z = 0$.

$$\begin{bmatrix} x_w \\ y_w \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ 1 \end{bmatrix} \tag{16}$$

where the camera coordinates (x_c, y_c) are mapped to the world coordinates (x_w, y_w) using the homography H. The RANSAC [50] algorithm is used to improve the estimation step due to its robustness to matching errors between the board corners and the points clicked on the image. The method computes the homography of random subsets of the corresponding points and calculates the number of inliers of the reprojected points. The subset with the most inliers is used to estimate the homography matrix. Regarding the orientation transformation, the y-axis from the BigMag reference frame has the same direction as the row axis of the image, while the x-axis and column have opposite directions. Therefore, the eq. 17 is used for the transformation of the angle of the robot from the image (α_c) to the BigMag reference frame (α_w).

$$\alpha_w = \tan^{-1} \frac{-\sin \alpha_c}{\cos \alpha_c} \tag{17}$$

2.3 Ultrasound-based tracking

In this section, we demonstrate the integration of an ultrasound machine into the BigMag system to localize and control the magnetically actuated soft robot to a target. Two different approaches are used for the

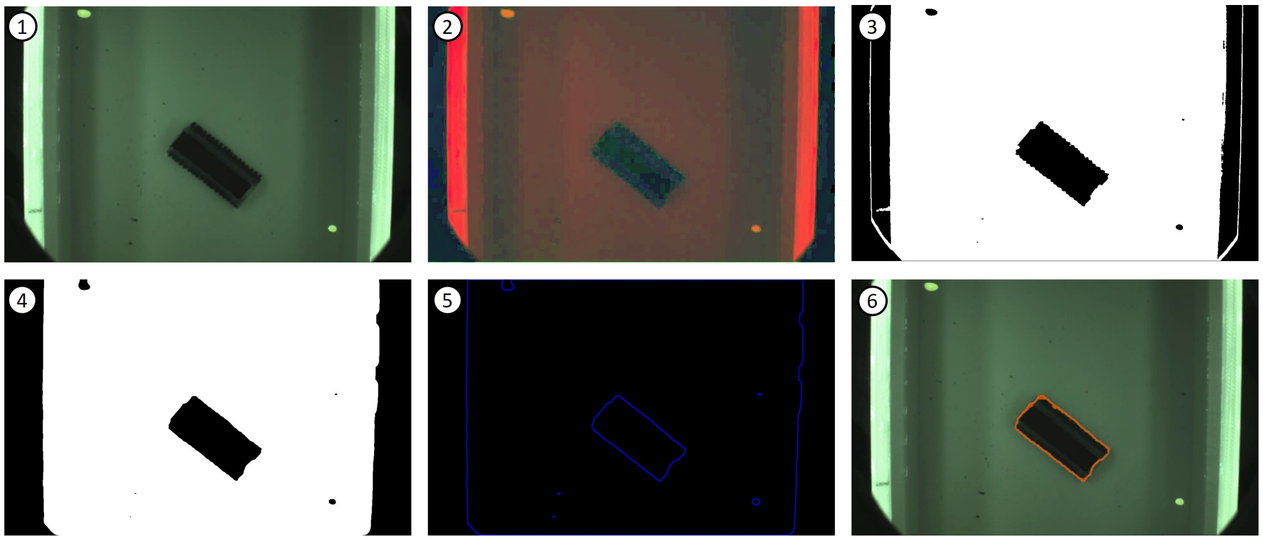


Figure 10: Computer vision applications for estimating the position of the robot. The original Red-Green-Blue (RGB) image (1) is converted to a Hue-Saturation-Value (HSV) image (2). (3) Adaptative threshold for obtaining a binary image (black and white). (4) Morphological opening, i.e. dilatation of the erosion, is applied to remove small blobs. (5) Edge detection to find the contours in the image. (6) The contour of the robot is selected according to the area and shape of the contour. Then the centroid and orientation is estimated using the moments of the contour.

real-time localization of the robot in an ultrasound image. The first approach utilizes a geometric algorithm to retrieve the shape of the robot, while the second approach uses a custom-made CNN.

2.3.1 Setup

Ultrasound imaging requires a medium to propagate the high frequency sound waves, so the soft robot is immersed in water inside an acrylic box, as shown in Fig. 11. The pipeline of the integration of the ultrasound Clarius probe into the BigMag system is pictured in Fig. 9. The Clarius probe acquires the ultrasound images of the workspace using a Clarius App on a smartphone and sends the images to a cast API on a laptop, as described previously. Since the Clarius Cast API is not compatible with the Linux Ubuntu 14.04 running on the computer of BigMag, the repository was implemented on a computer running Linux Ubuntu 20.04, equipped with Intel i7-8750H CPU, GeForce GTX 1050 GPU and 16 GB of RAM. Then, the computers were connected via an Ethernet cable to establish a UDP communication protocol. The new modified Cast API captures the ultrasound images, applies a localization algorithm, and sends the position of the robot to the computer of BigMag. Moreover, for each estimation, the Cast API creates a buffer of characters, containing the protocol code and the three values of the pose of the Millipede. Then, the BigMag controller loop receives the buffer and converts the values back to double variables [63]. At the same time, the ultrasound image and the respective label were saved in the laptop. Finally, the control and magnetic actuation system run similar to the camera-based tracking previously implemented.

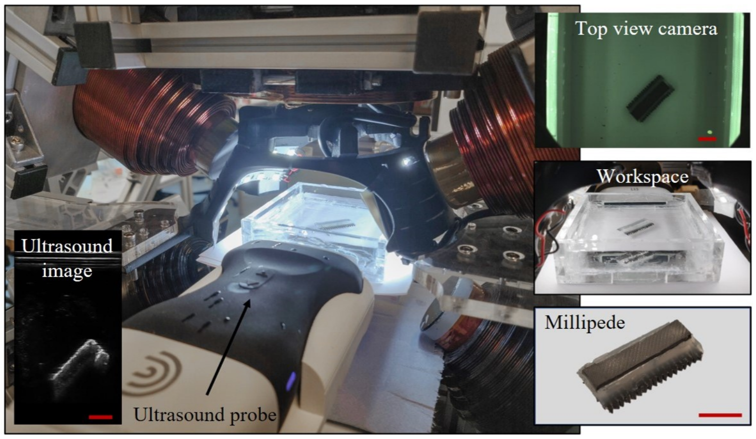


Figure 11: An ultrasound probe is placed in the workspace of BigMag to allow ultrasound imaging of the robot. Left: Overview of the BigMag workspace with ultrasound probe. Right: ① water tank, ② Millipede robot, ③ probe holder, ④ Clarius ultrasound probe.

2.3.2 Geometric algorithm

When controlling the robot through the reachable workspace, the two perpendicular edges of the robot are visible most of the time in the ultrasound images. So, the geometric approach relies on finding the edges of the Millipede to estimate the pose of the robot. The approach is a two stage algorithm to first perform image segmentation and then detect and estimate the position and orientation of the soft robot, as shown in Fig. 13.

The first stage is like the camera-based algorithm described previously, but the ultrasound image is in the greyscale colour space. Also, the ultrasound image is noisier which requires additional filtering, such as a median filter to remove salt and pepper noise, and a Gaussian filter to remove speckle noise and smoothen the image. Then, the image is binarized with the adaptative threshold algorithm. Morphological opening is applied to remove small blobs, while keeping the shape of the robot. Afterwards, the contours of the binary image are extracted, using the Canny edge detector [49]. The size of the median filter, the Gaussian filter, and the morphological opening are defined by the user.

The second stage is the object detection algorithm, which searches a rectangular shape on the image of the edge points. The edge points are candidates to fit the two perpendicular lines of the rectangular shape of the Millipede. A general GPU implementation of the RANSAC is modified to fit a line to the set of points

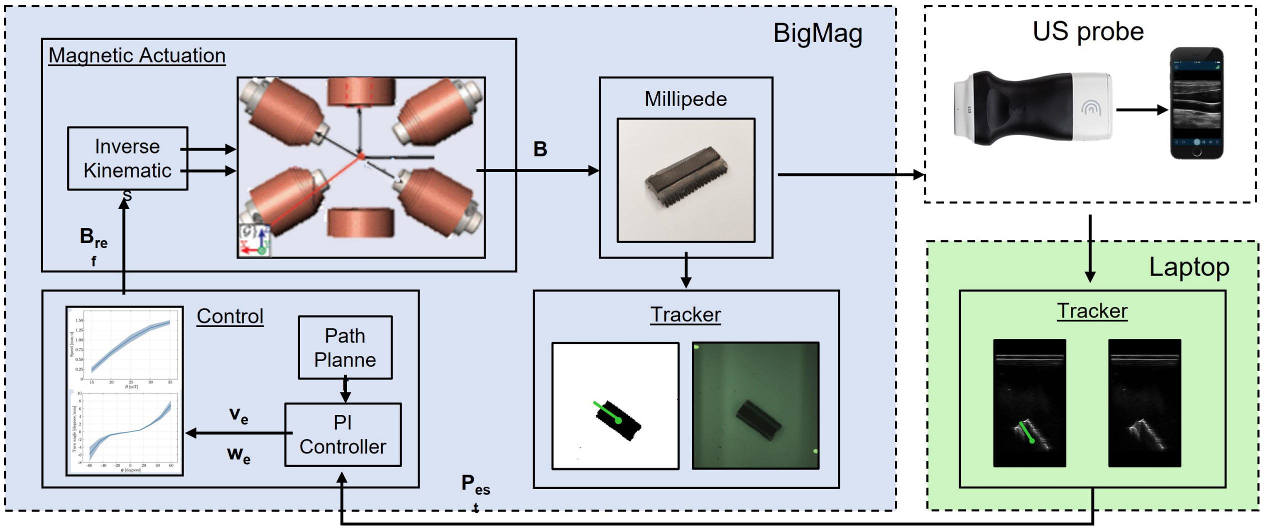


Figure 12: Block diagram summarizing the closed-loop system for controlling magnetic soft robots using ultrasound imaging. The blocks of the magnetic actuation and control are identical to the pipeline using the camera, but the tracking system uses ultrasound. The ultrasound machine images the Millipede and a tracking algorithm estimates the pose of the soft robot (P_{est}). The pose of the robot is sent to the PI controller through a UDP connection between the laptop and the BigMag system. The camera tracker defines the ground truth of the pose of the Millipede, and it is not connected to the control block.

and save the best candidate lines. Then, the slope of the perpendicular line is calculated for each candidate line and inserted into the second iteration of RANSAC. It helps to discard candidate points to the line fitting and, thus, saving computation time. The sum of inliers of the candidate line and its perpendicular are used to choose the best pair. The length of each line is calculated by the distance between the intersection point and the furthest inlier along the line. The slope of the bigger edge sets the orientation of the robot (α_{robot}). Then, the bigger and smaller edges define the corner of the rectangle detected, which is used to calculate the position of the robot. The corner is in relation with the coordinate frame of the image, i.e. columns and rows. For instance, if the corner detected is the bottom left (BL) corner, as shown in Fig. 13, the formula of the centroid (x_{robot}, y_{robot}) is

$$x_{robot} = x_{BL} + \frac{w}{2} \cos\alpha_{robot} + \frac{h}{2} \sin\alpha_{robot} \tag{18}$$

$$y_{robot} = y_{BL} + \frac{w}{2} \sin\alpha_{robot} + \frac{h}{2} \cos\alpha_{robot} \tag{19}$$

where w and h are the width and height of the robot in pixels, respectively. Then the coordinates are mapped to the BigMag coordination frame, using a homography from the calibration of the ultrasound image.

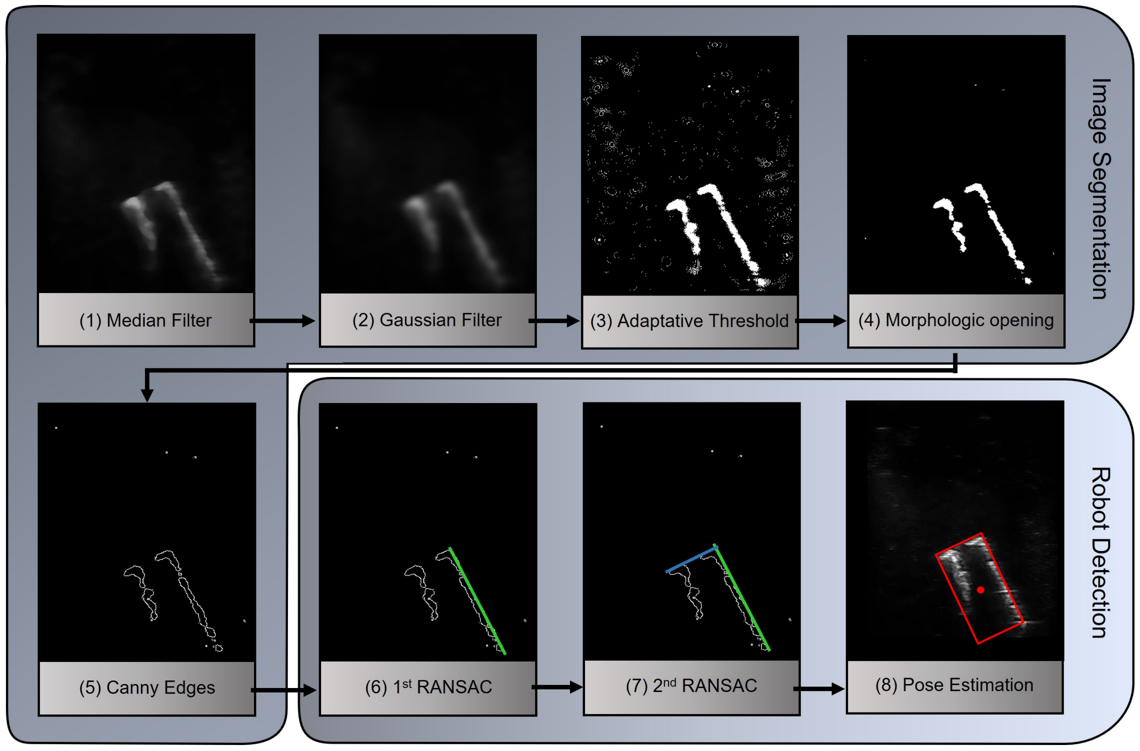


Figure 13: Computer vision applications for estimating the position of the robot using ultrasound images. The first stage consists of image segmentation (1) - (5) followed by the detection of the robot (6) - (8). (1) (2) Image preprocessing of the ultrasound image for speckle noise reduction and for image smoothing. (3) Adaptative threshold to obtain a binary image (black and white). (4) Morphological opening, i.e. dilatation of the erosion, is applied to remove small blobs. (5) Detection of the candidate points for estimating the pose of the robot, using Canny edges. (6) First RANSAC iteration to find the principal edge of the soft robot. (7) Second iteration of RANSAC to find the perpendicular edge of the robot. (8) Pose estimation by calculating the centroid and the orientation of the robot.

2.3.3 Deep Learning

The second approach uses a convolutional neural network to estimate the position and orientation of the robot in the ultrasound image. The process of acquiring the ultrasound images and the label for the dataset is described. The architecture of the proposed CNN and the training of the network is also presented.

DataSet The dataset was constructed with ultrasound images and the pose of the soft robot. To avoid annotating all the video frames manually, we used the pose estimation from the camera-based algorithm to label each ultrasound image. Therefore, the ultrasound images were received from the Clarius probe and the respective label were received from the camera-based algorithm. The pipeline is identical to the pipeline of Fig. 12, but the camera-based tracker provides the pose of the robot to the control block and transmits the pose values to the laptop block. Furthermore, for each estimation, the BigMag tracker block creates a buffer of characters, containing the protocol code and the three values of the pose of the Millipede. Then, the modified Cast API running on the laptop receives the buffer and converts the values back to double variables. The ultrasound images were stored with the correspondent ground truth values of the position (x_0, y_0) and the orientation α_0 of the robot. Since the ultrasound acquisition rate (16 Hz) is higher than the BigMag control rate (10 Hz), the ultrasound images are saved at 10 Hz rate, when a new pose of the robot is received.

The creation of the dataset consisted of navigating the robot through the workspace reachable by the ultrasound machine. It was important to have a well-balanced dataset, representing a wide range of possible positions and orientations of the robot. So, the robot was controlled through the workspace and rotated over its axis to gather the maximum combinations of positions and orientations.

Furthermore, the dataset was augmented to increase the robustness and the size of the dataset. Therefore, each image was filtered either with a Gaussian filter or with a median filter to avoid some biases of the convolutional neural network. The median filter increases the salt and pepper noise of the ultrasound image, while the Gaussian filter smooths the image, removing some artefacts.The dataset is constituted of approximately 95000 ultrasound images and the respective labels. The dataset was divided into train(80%), validation(10%) and test(10%).

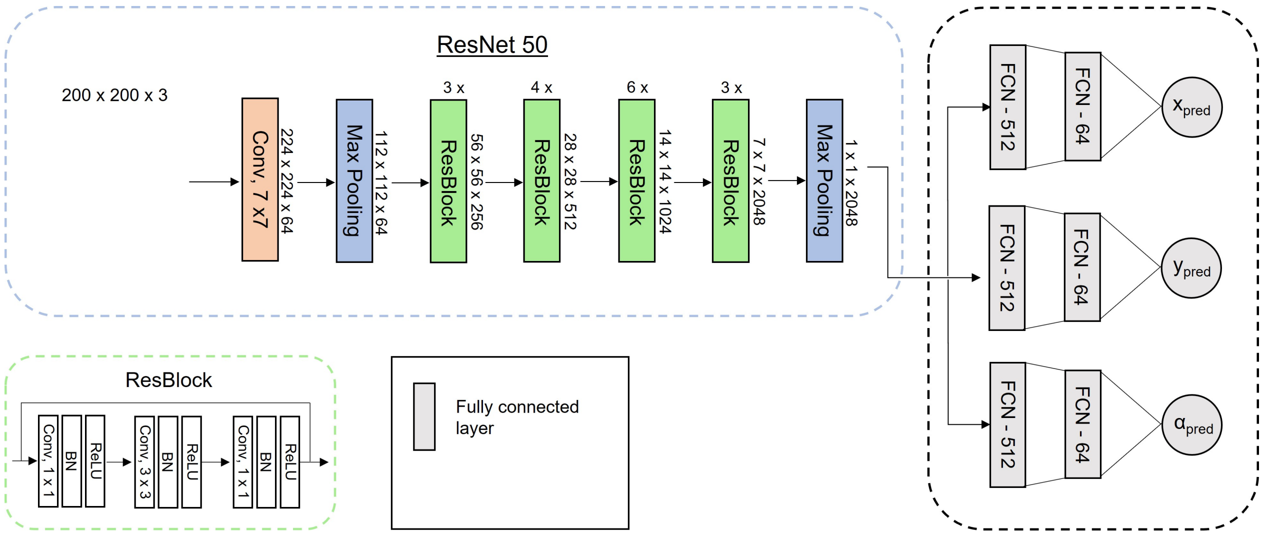


Figure 14: Architecture of the adapted ResNet 50 convolution neural network (CNN) for pose regression. The CNN proposed uses a ResNet-50 as backbone for feature extraction. The features image vector is used in three separate branches for regression of the pose (x_{pred} and y_{pred}) and the orientation (α_{pred}) of the robot. Each branch is composed by three consecutive fully-connected layers with 512, 64 and 1 outputs.

Architecture The proposed CNN uses a ResNet-50 as backbone to estimate the position and orientation of the robot. The input of the network is a greyscale image with a fixed size of 200x200 pixels. The output of the network is a 3x1 vector with the predicted pose of the robot $(x_{pred}, y_{pred}, \alpha_{pred})$, as shown in Fig. 14.

The ResNet-50 is a 50-layer deep residual network which has five convolutional blocks stacked on top of each other. The output of the last convolutional block is a 7x7x2048 dimensional array which is flattened to a 2048-dimensional vector using a max pooling layer. The original ResNet-50 used in ImageNet [19] dataset uses a Fully Connected layer with 1000 output-neurons (FCN-1000) to extract features to classify the 1000 classes of the ImageNet dataset. So, the FCN-1000 layer is removed and the 2048-dimensional vector is fed into three regression branches. Each branch performs a regression of one of the pose values, where each is composed by a set of 3 fully connected layers, with 512, 64 and 1 output neurons. The first two fully connected layers use a ReLU activation function, while the last uses a sigmoid function to predict the parameters of the robot pose. Finally, each value of each regression branch is concatenated into a 3x1 vector. The branches for the regression of x and y component utilize a root mean squared error loss function (RMSE) to compute the minimum distance between the predicted and labelled value, y and $\tilde{y}$, respectively. On the other hand, the regression of the angle uses a modified mean squared error function to handle the circular angle representation. The MSE penalizes larger errors, such as the pair (1°; 359°), despite being very similar. So, the angle regression loss function changes the mean squared error (CMSE) between the angles by representing the angle in the complex plane using $z(y) = (cos(y), sin(y))$.

$$L_{RMSE} = \sqrt{\frac{1}{N}\sum_{n=1}^{N}(y_i - \tilde{y}_i)^2} \tag{20}$$

$$L_{CMSE} = \frac{1}{N}\sum_{n=1}^{N}\left|z(y_i) - z(\tilde{y}_i)\right|^2 = \frac{1}{N}\sum_{n=1}^{N}(\cos(y_i) - \cos(\tilde{y}_i))^2 + (\sin(y_i) - \sin(\tilde{y}_i))^2 \tag{21}$$

Training The training process uses a normalization of the dataset constructed. The ultrasound images are rescaled to the input dimension of the neural network proposed. The rescaled methods are shown in Fig. 15. The first method rescales the original image from a size of 640x360 pixels directly to the input size of 200x200 pixels, using a cubic interpolation. This approach resizes the width at a different rate when compared with the length of the image, 1.8:1 and 3.2:1, respectively. The second approach consists of adding black pixels to the original image to create an image with 640x640 pixels size. The image width and length are resized at the same rate 3.2:1, using cubic interpolation. The second approach is used to resize the images of the dataset to keep the aspect ratio. The labels of the dataset are also normalized. The orientation angles are remapped to the interval 0 to 180 degrees, since the robot can move in both directions. Then the angle values are normalized by dividing the labelled value by 180.

The ResNet-50 implementation is initialized with the weights from the pre-trained residual network on the ImageNet dataset [19]. The weights of the rest of the proposed CNN are randomly initialized. During the training phase, the dataset is divided into training (80%), validation (10%), and testing (10%). As optimizer, the Adam algorithm [68] is used with the default values and with a learning rate of 0.01.

Fig. 16 shows the x-component, y-component, angle, and the total losses for each iteration of the training and validation. The loss values decrease for all estimated pose values. Also, overfitting is avoided, since the validation loss is smaller than the training loss.

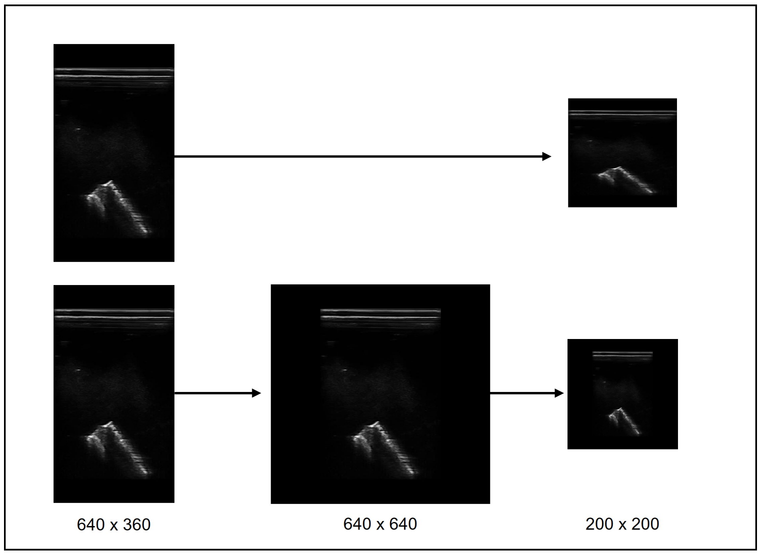


Figure 15: Two image resize approaches. The first approach (top) directly resizes the original image to the input size of the CNN, resulting in distortion. The second approach (bottom) adds black pixels to the original image to equalize the resize rate between the columns and rows.

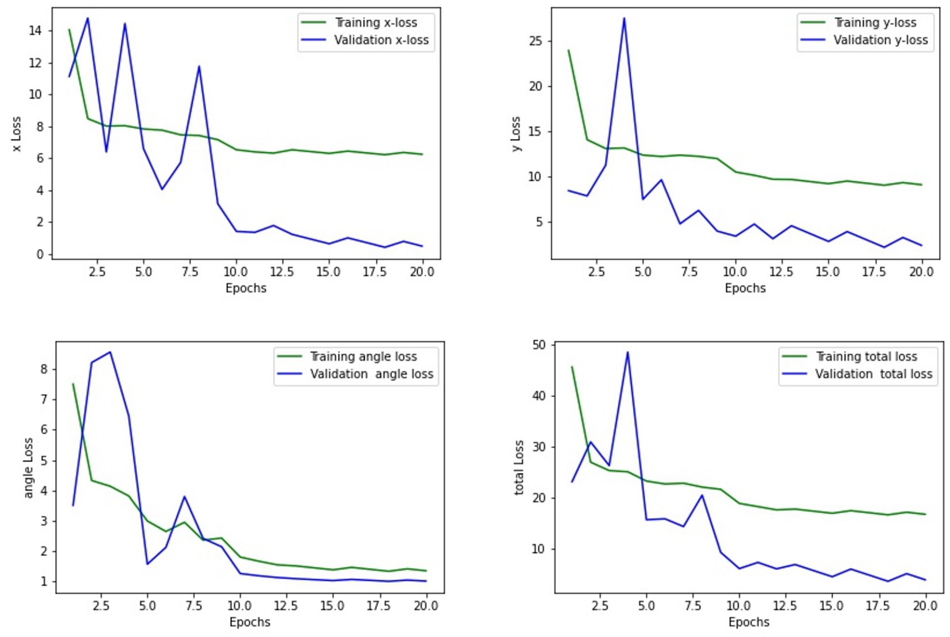


Figure 16: Plots of the losses of the validation and training for each iteration. Top-left: validation and training losses of the x-component for each iteration. Top-right: validation and training losses of the y-component for each iteration. Bottom-left: validation and training losses of the angle for each iteration. Bottom-right: total loss of the validation and training for each iteration.

3 Results

In this section, we demonstrate the closed-loop control of untethered soft robots, using ultrasound feedback. The camera tracking and control is validated as proof-of-concept for the autonomous navigation of the Millipede robot. The ultrasound tracking methodologies proposed for the estimation of the pose of the robot are compared. Finally, we evaluate the performance of the closed-loop control of the Millipede, using ultrasound tracking.

3.1 Camera tracking

The camera-based algorithm is considered the ground truth position of the Millipede robot. So, the closed-loop control of the Millipede to reach a target is evaluated to validate the control strategy proposed to navigate the robot. The workspace has two virtual obstacles in right and left side, so that the steering of the robot to avoid obstacles in either side could be validated. Fig. 17 shows snapshots of the closed-loop navigation of the Millipede using camera feedback, while avoiding obstacles. The path search is initialized at the position of the robot, while the target is selected by the user. The resultant waypoints of the path are used in the control of the Millipede to the target. The actuated magnetic field generates displacement of the legs of the robot, which can result in a straight-line or turning motion. The kinematics is similar to the differential drive of a two-wheel mobile platform and some parameters have to be tuned. The control gains of the control equations are empirically chosen to allow driving stability. Also, the control distance of eq. 12 is set to half of the length of the robot to increase the stability of the driving motion.

The trajectory of the soft robot when following the waypoints of the path planner is shown in Fig. 18. The average speed of the robot is $1.61\,\mathrm{mm\,s^{-1}}$, which the average speed can be increased by increasing the control gains and, consequently, the intensity of the electromagnetic field. The mean error of the deviation of the soft robot from the path planner is $1.71 \pm 1.07\,\mathrm{mm}$, which corresponds to approximately 15% of the width of the Millipede. The result is acceptable, since the differential drive does not allow the robot to move perpendicular to the legs' direction to correct the error.

The results of the closed-loop navigation of the untethered soft robot, using camera feedback, validate the control strategy to steer the robot. Also, the path planner is capable of providing the necessary waypoints to avoid the collision of the robot with obstacles.

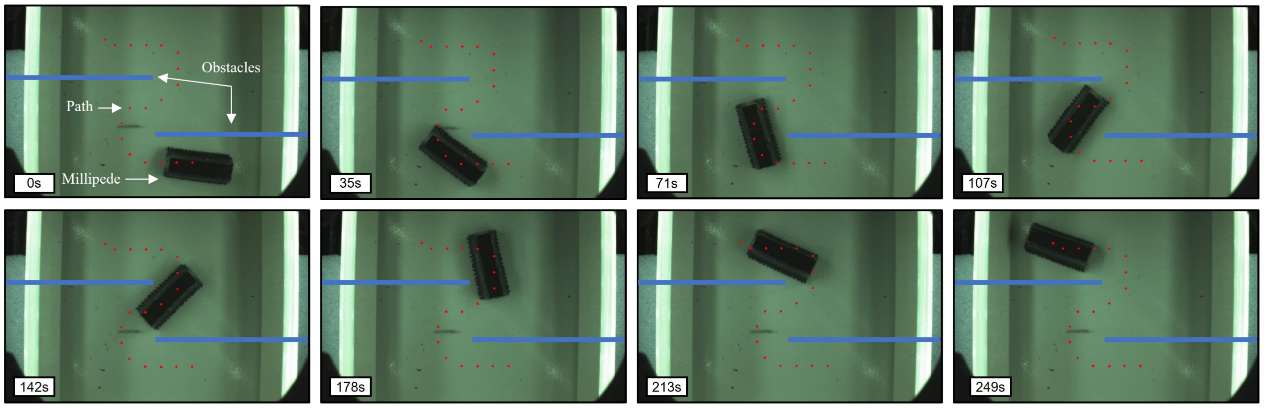


Figure 17: Snapshots of the closed-loop navigation of the Millipede using camera feedback. The robot follows a path that avoids the obstacles of the workspace to reach the target site selected by the user.

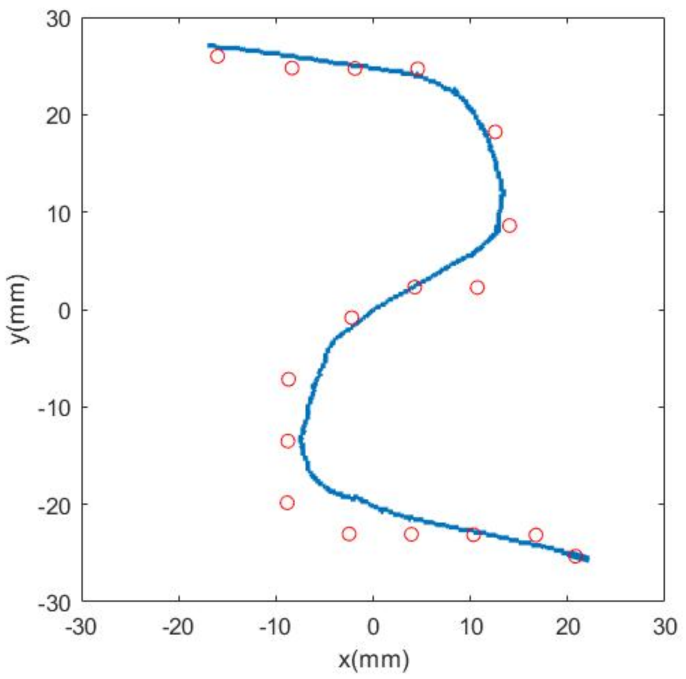


Figure 18: Motion control of the Millipede robot based on the feedback provided by the camera. Way points (red circles) are provided to the control system to define the trajectory. The average speed of the robot is 1.61 m s^{-1}. The mean error of the deviation of the soft robot from the path planner is 1.71 ± 1.07 mm.

3.2 Ultrasound tracking

The performance of the two ultrasound-based methods to estimate the pose of the robot is evaluated. The performance is evaluated in a test dataset of ultrasound images labelled with the ground-truth poses given by the camera. Then, the best ultrasound tracking method is used to control the Millipede to a target.

First, we evaluate the performance of both tracking methodologies in the test dataset. The test dataset is composed by 9500 ultrasound images and the respective label, and it corresponds to 10% of the entire dataset. The test dataset is composed by ultrasound images never seen by the CNN during the training.

The position tracking error is evaluated in the four image quadrants to evaluate the performance of the methods in different regions of the ultrasound image, as shown in Table 1. Overall, the error between the predictions of both methods with the ground truth is higher in the y-component than the x-component in all four quadrants. Furthermore, the performance of the CNN is better than the RANSAC based method, since the absolute error and the standard deviation are smaller in every quadrant. The performance of the RANSAC method is constant across the quadrants, despite the higher mean absolute error. Regarding the CNN method, Also, regarding the average error of the CNN tracking, when the robot is further from the ultrasound probe ($y > 0$), the error in the y-component is 2-3 times bigger than the error in the other half of the workspace. On the other hand, the error of the x-component remains constant.

Table 1: Average tracking error of the RANSAC and CNN methods with respect to the 4 Quadrants.

Coordinates	Data	Axis	RANSAC (mm)	CNN (mm)
x>0 & y>0	2314	x	0.7 ± 1.2	0.2 ± 0.1
		y	0.9 ± 0.9	1.0 ± 2.0
		Absolute	1.14 ± 1.5	1.0 ± 2.0
x>0 & y<0	3155	x	1.0 ± 1.6	0.2 ± 0.1
		y	1.4 ± 1.8	0.4 ± 0.3
		Absolute	1.7 ± 2.4	0.5 ± 0.3
x<0 & y>0	2550	x	0.9 ± 0.9	0.2 ± 0.2
		y	1.1 ± 0.8	0.7 ± 1.0
		Absolute	1.4 ± 1.2	0.7 ± 1
x<0 & y<0	1483	x	1.3 ± 2.1	0.2 ± 0.2
		y	1.9 ± 4.1	0.4 ± 0.2
		Absolute	2.3 ± 4.6	0.5 ± 0.3

The test dataset was divided into intervals of 20° according to the angle labelled in each ultrasound image. The absolute error between the ground-truth and the predicted angle by both methodologies is calculated. Fig. 19 shows the dispersion of the mean angle for each angle interval. Overall, the dispersion of the error is lower in the CNN method, except for the intervals near the angle discontinuity. The RANSAC based method performs badly, when the robot is facing the ultrasound probe, because only the small edge of the robot is imaged. The estimation of the pose of the robot with the CNN ultrasound method is better and more reliable, since the error and the standard deviation are 2 times smaller than the estimations by the RANSAC method.

The performance of the CNN tracking method is better for the estimation of the position in all four quadrants of the image. It is also better for the prediction of the orientation of the robot for most of the angle intervals. The mean results for the estimation of the pose also confirm the higher performance of the CNN method. For the RANSAC based method, the tracking errors for the x-component, y-component and angle are 0.9 ± 1.4 mm, 1.2 ± 2.1 mm, and 3.9 ± 6.7°, respectively. On the same dataset, the proposed CNN obtains errors of 0.28 ± 0.14 mm, 0.62 ± 0.48 mm, and 2.3 ± 3.4° for the x-component, y-component, and the angle, respectively. Thus, the proposed CNN is more reliable for the estimation of the position and orientation of the robot, using ultrasound images. The proposed CNN tracking method is selected to estimate the pose of the Millipede in the closed loop-control, using ultrasound images.

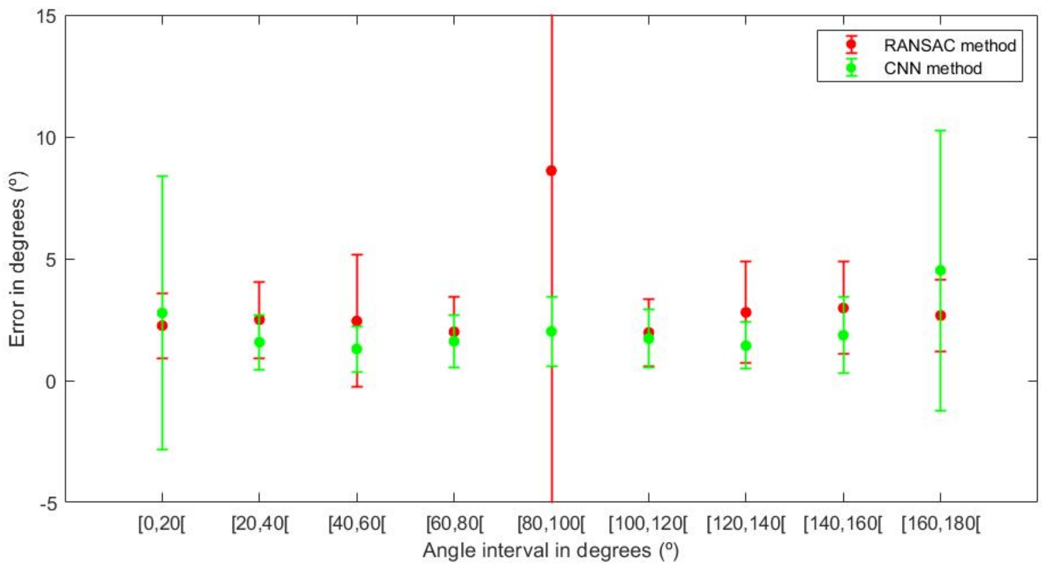


Figure 19: Absolute error between the predicted angle by the proposed tracking strategies and the ground-truth angle. The mean error and the respective standard deviation are calculated for each 20° interval containing the ultrasound images, with the ground truth in the correspondent interval.

The previous comparison gives an overview of the performance of both methods in different positions and orientations of the workspace. However, the comparison do not provide the perception of the tracking error, during the continuous navigation of the robot. Therefore, we compare the error of the estimation of the position and the pose for both methods, during the control of the soft robot through the workspace. The control is obtained with the camera-based tracker, and we navigate the robot using the motion commands (forward, backwards and turning).

Fig. 20 shows the trajectory performed by the Millipede robot and the comparison between the pose estimated by the camera tracking and both ultrasound tracking methods, the CNN and the RANSAC. Initially, until the 90 s mark, the robot was moved forward, turned and moved backwards. Also, the trajectory shows a 180° rotation of the robot about its axis (from 90 s to 130 s). For the CNN based method, the absolute and angle tracking errors are 0.5 ± 0.4 mm, and 1.5 ± 2.0°, respectively. The errors are comparable to the errors obtained in the experiments of the estimation of the pose of the Millipede in the ultrasound images dataset. On the other hand, the proposed RANSAC obtains errors of 1.4 ± 0.9 mm, and 3.1 ± 8.2° for the absolute distance and the angle, respectively. Despite the acceptable mean values of the tracking error of the RANSAC based method, the tracking error when the robot is closed to an orientation of 90º is huge. The moments are visible in the plot of the angle in Fig. 20, during the time interval near the 60 s and 150 s. Furthermore, when the angle of the robot is between 80° and 100°, the orientation error was 7.9 ± 20.9° and the error of the position was 1.9 ± 1.5 mm. In these situations, the robot is facing the ultrasound probe and thus, only the small edge of the robot is visualized, which increases the difficulty of regression the pose of the robot. This type of behaviour limited the application of the RANSAC algorithm to the closed-loop system. Experiments showed that the robot could not turn, once the robot was perpendicular to the probe. A solution might be to keep applying the same magnetic field until the detection of the orientation is more stable. However, this would decrease the reliability of the tracking and control system to perform path following experiments.

Thus, the CNN pose estimation presents a lower and more stable position and angle estimation, and it was used in the closed loop control of the diagram of Fig. 12. Since the CNN was trained in python, the CNN architecture and the weights of the layers are stored and converted to be used in the C++ Cast API on the laptop. The Deep Neural Networks module of OpenCV allows the use of the trained CNN in the c++ environment [62]. The estimation by the CNN runs at 15 Hz on the laptop computer. The CNN estimates the position and orientation of the Millipede for each frame acquired by the Cast API. Then, the information is sent to the control system implemented in BigMag and the required magnetic field is applied, as described in Fig.

The previous experiment focused only on the performance of the CNN and RANSAC methods to estimate the pose of the Millipede. However, the total error of the closed-loop system incorporates the tracking error previously estimated, the error associated to the motion control and the error of the generation of the magnetic field. Therefore, we study the error of the closed-loop system using each ultrasound tracking method. Since the ultrasound probe images a smaller area, when compared to the top view camera, the robot could not perform the same path as in the camera tracking experiment. Thus, two separated experiments are performed, while avoiding an obstacle on the left and other on the right to show the ability of steering the robot in both directions. Fig. 21 shows snapshots of the closed-loop navigation of the Millipede using ultrasound feedback, while avoiding obstacles. The results obtained are the mean values of 2 identical experiments.

The experiments demonstrate the closed-loop control of the Millipede, using ultrasound imaging feedback, are performed with the presence of an obstacle on one of the sides of the workspace. Figs. 22, 23 show the trajectories produced by the untethered soft robot, when avoiding an obstacle on the right and on the left, respectively. The error of the ultrasound trajectory is obtained by the absolute Euclidean distance between the estimated poses by the camera tracker and by the CNN ultrasound method. Moreover, the angle difference, which is plotted in Fig. 24, is obtained by the absolute angle difference between the angles estimated by the camera tracker and the CNN ultrasound methos. For the path that avoids the obstacle on the right, C-shaped path, the absolute tracking error is 1.71 ± 0.85 mm and the angle error is $2.29 \pm 1.64°$. For the other path, D-shaped path, the absolute tracking error is 1.49 ± 0.83 mm and the angle error is $2.18 \pm 1.33°$.

The mean deviation from the CNN estimation to the trajectory of the path planned is 1.87 ± 0.92 mm, which corresponds to approximately 16% of the width of the Millipede. The result is similar to the mean deviation of the robot, when using camera-feedback, which proves the ability of the untethered robot to follow a path, using ultrasound imaging feedback.

The closed-loop experiments showed that the robot is capable of following a path to a target, using ultrasound feedback. Nevertheless, the camera tracking algorithm shows that the robot keeps a stable trajectory to the target, despite the errors of the estimation of the pose of the robot during the experiment. Therefore, the control system is capable of steering the robot towards the waypoints of the path, even in the presence of occasional errors. The control system can tolerate small errors due to some possible reasons. There is a delay between the commands to move the robot and the displacement of the legs of the soft robot. It allows an incorrect estimation because the soft robot do not respond immediately to the application of the misplaced magnetic field. The other possible reason is that the soft nature of the robot and the actuation of a rotational magnetic field allow that the point of application can be deviation from the centroid of the robot. The movement is not as clean, but it moves the robot in the desired direction.

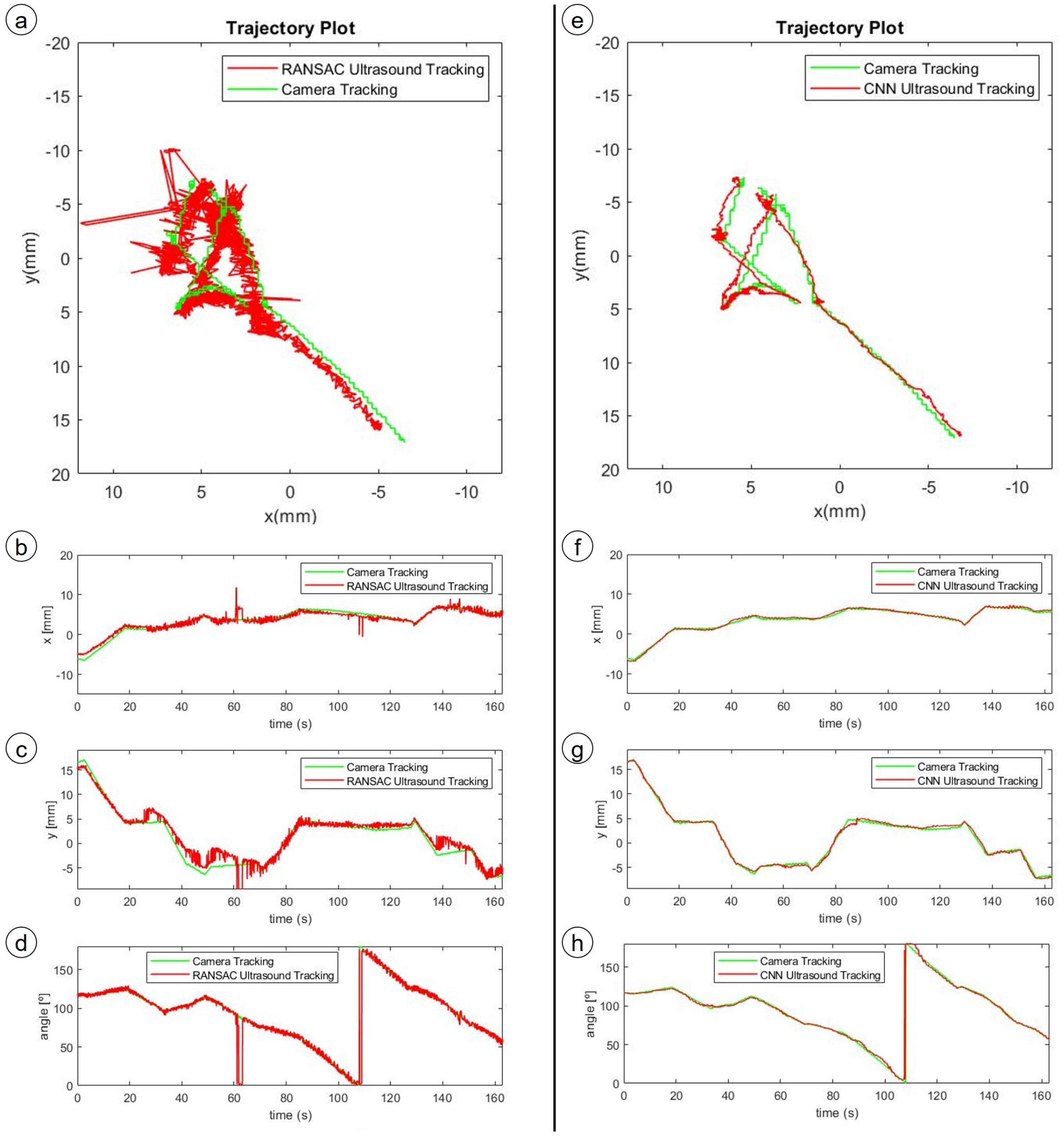


Figure 20: Comparison between the poses estimated by the CNN and RANSAC methodologies, during the navigation of the robot. ⓐ - ⓓ correspond to the trajectory, the x-component, y-component and the angle plots of the estimation from the RANSAC method in ultrasound images and camera methods in camera frames. ⓔ - ⓗ correspond to the trajectory, the x-component, y-component and the angle plots of the estimation from the CNN method in ultrasound images and camera methods in camera frames.

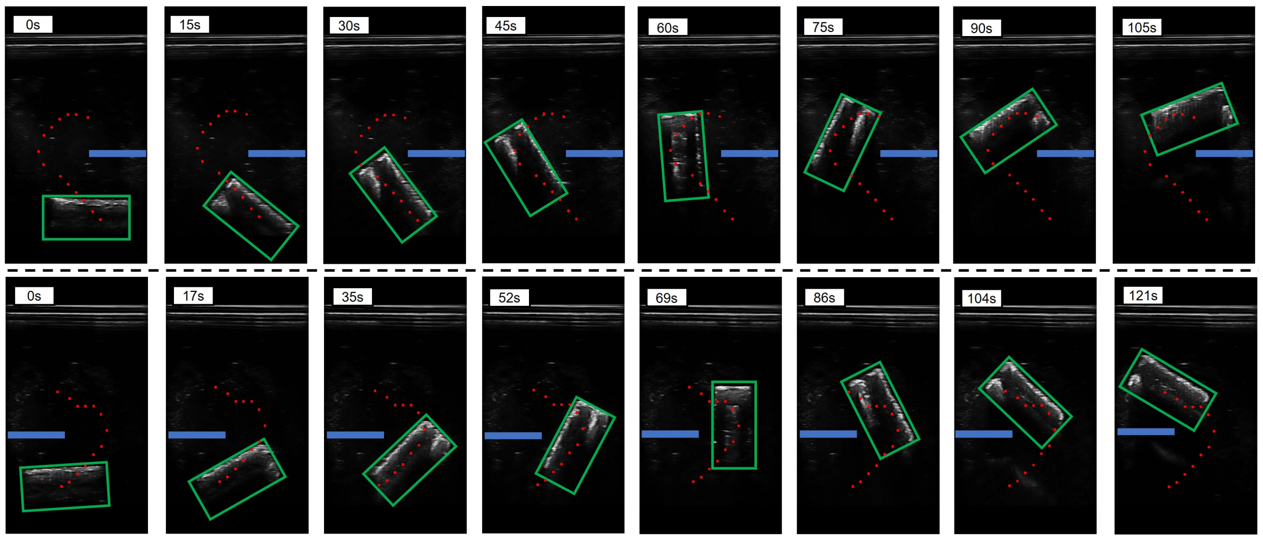


Figure 21: Validation of the closed-loop control of a soft robot using a convolutional neural network to detect the pose of the robot in ultrasound images. The experiments show the steering of the Millipede robot to a target while avoiding an obstacle on the right (top row) and on the left (bottom row).

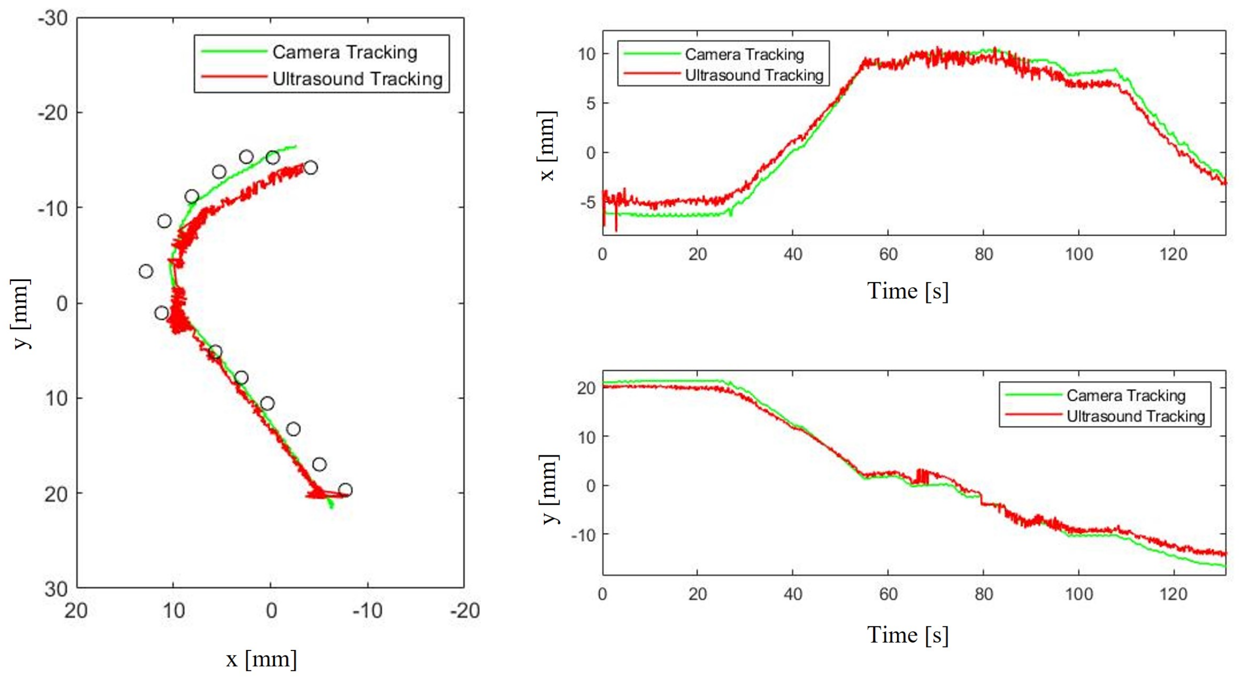


Figure 22: Motion control of the Millipede based on the feedback provided by the ultrasound system and the CNN pose estimator, while avoiding an obstacle on the left. Left: Trajectory of the robot, using ultrasound imaging, and the real trajectory of the robot given by the camera imaging. Way points (black circles) are provided to the control system to define the trajectory. Right: Plots of the evolution of the x-component (top) and y-component (bottom) vs time.

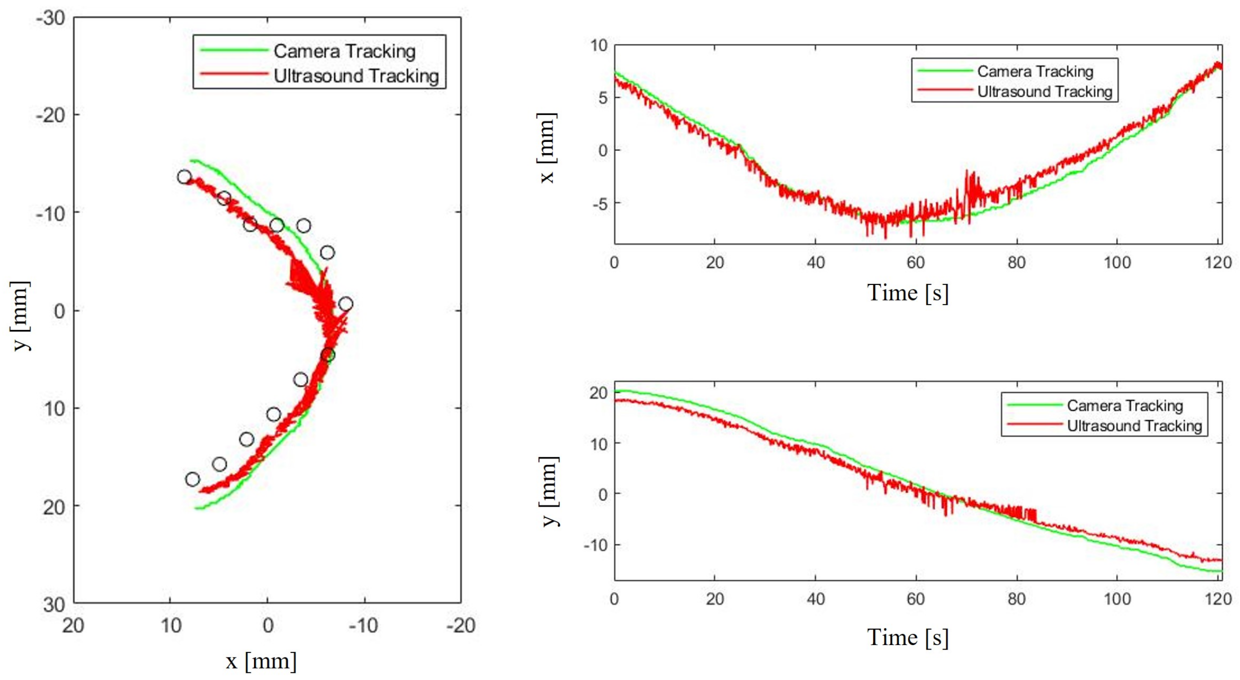
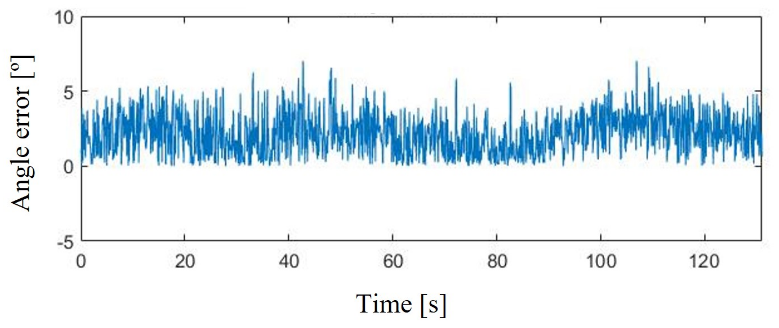


Figure 23: Motion control of the Millipede based on the feedback provided by the ultrasound system and the CNN pose estimator, while avoiding an obstacle on the left. Left: Trajectory of the robot, using ultrasound imaging, and the real trajectory of the robot given by the camera imaging. Way points (black circles) are provided to the control system to define the trajectory. Right: Plots of the evolution of the x-component (top) and y-component (bottom) vs time.



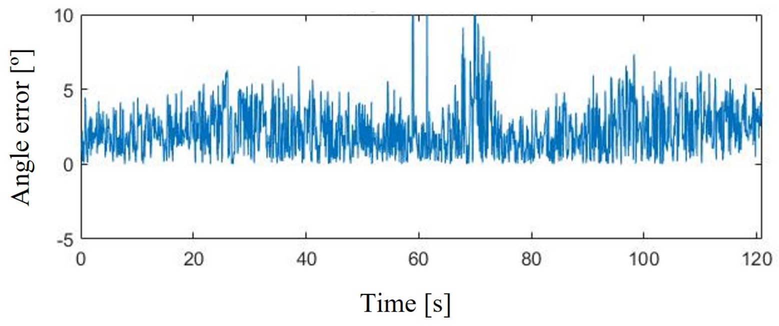


Figure 24: Plots of the absolute angle error during motion control of the Millipede, using CNN ultrasound tracking. The top and bottom plot correspond to the C-shaped, Fig. 22, and D-shaped, Fig. 23, trajectories, respectively.

Chapter 3 (Discussion)

This thesis demonstrates the closed-loop control of an untethered soft robot, using ultrasound imaging feedback. The key features of the work can be summarized in:

- Integration of an ultrasound probe into an actuation magnetic system with six mobile electromagnetic coils.
- Development of a convolutional neural network (CNN) capable of estimating the position and orientation of the Millipede robot in ultrasound images.
- Closed-loop navigation of the untethered soft robot to a target, while avoiding obstacles, using ultrasound imaging feedback.

In summary, the results demonstrate good performance of the CNN method over a geometric method to estimate the pose of the robot in a dataset containing more than 9500 ultrasound images. The average tracking error for the position and orientation is 0.68 ± 0.50 mm and $2.3 \pm 3.4°$, respectively. Furthermore, the results of the closed-loop control of the Millipede, using the CNN tracking in the ultrasound images, show a tracking error of 1.6 ± 0.84 mm and an angle error of 2.23 ± 1.48 mm. The deviation of the robot from the path planned is 1.87 ± 0.92 mm.

The performance of the ultrasound tracking and control can be further enhanced by adding some improvements to the system. The estimation of the pose of the robot by the proposed CNN needs to be processed before reaching the control diagram. The Extended Kalman Filter [69] can be used to improve the performance of the tracking strategy, especially for the ultrasound frames that the CNN misses the estimation. Secondly, the path planner should include the planning of the motion of the robot through the workspace. The RRT allows estimating an obstacle free path as well as the commands necessary to steer the robot to the target. This strategy is useful to output a path that is feasible for the kinematics of the robot.

For future work, the ultrasound tracking and control can be adapted to other type of magnetically actuated soft robots. Also, the Millipede robot can be improved: miniaturization to allow a higher manoeuvrability, and improvement of the motion on non-horizontal surfaces. This can open the possibility to control the Millipede in a 3D space. Thus, the proposed tracking system needs modifications to allow the estimation of the 6D pose of the robot in the 3D space. The 3D space can be imaged in real-time using ultra-fast ultrasound images that sweeps the workspace or using a 3D ultrasound probe.

References

[1] Metin Sitti, Hakan Ceylan, Wenqi Hu, Joshua Giltinan, Mehmet Turan, Sehyuk Yim, and Eric Diller. Biomedical applications of untethered mobile milli/microrobots. *Proceedings of the IEEE*, 103(2):205–224, 2015.

[2] Gavriel Iddan, Gavriel Meron, Arkady Glukhovsky, and Paul Swain. Wireless capsule endoscopy. *Nature*, 405(6785):417–417, 2000.

[3] Ian Wilding, Peter Hirst, and Alyson Connor. Development of a new engineering-based capsule for human drug absorption studies. *Pharmaceutical science & technology today*, 3(11):385–392, 2000.

[4] Marco Quirini, Robert J Webster, Arianna Menciassi, and Paolo Dario. Design of a pill-sized 12-legged endoscopic capsule robot. In *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pages 1856–1862. IEEE, 2007.

[5] Mingyuan Gao, Chengzhi Hu, Zhenzhi Chen, Honghai Zhang, and Sheng Liu. Design and fabrication of a magnetic propulsion system for self-propelled capsule endoscope. *IEEE Transactions on Biomedical Engineering*, 57(12):2891–2902, 2010.

[6] Levin J Sliker and Gastone Ciuti. Flexible and capsule endoscopy for screening, diagnosis and treatment. *Expert review of medical devices*, 11(6):649–666, 2014.

[7] Jo F Rey, Haruhiko Ogata, Naoki Hosoe, K Ohtsuka, N Ogata, K Ikeda, H Aihara, I Pangtay, T Hibi, S Kudo, et al. Feasibility of stomach exploration with a guided capsule endoscope. *Endoscopy*, 42(07):541–545, 2010.

[8] Gastone Ciuti, Pietro Valdastri, Arianna Menciassi, and Paolo Dario. Robotic magnetic steering and locomotion of capsule endoscope for diagnostic and surgical endoluminal procedures. *Robotica*, 28(2):199–207, 2010.

[9] Sehyuk Yim and Metin Sitti. Design and rolling locomotion of a magnetically actuated soft capsule endoscope. *IEEE Transactions on Robotics*, 28(1):183–194, 2011.

[10] Wenqi Hu, Guo Zhan Lum, Massimo Mastrangeli, and Metin Sitti. Small-scale soft-bodied robot with multimodal locomotion. *Nature*, 554(7690):81–85, Jan 2018.

[11] Venkatasubramanian Kalpathy Venkiteswaran, Danica Kristina Tan, and Sarthak Misra. Tandem actuation of legged locomotion and grasping manipulation in soft robots using magnetic fields. *Extreme Mechanics Letters*, 41:101023, 2020.

[12] Haojian Lu, Mei Zhang, Yuanyuan Yang, Qiang Huang, Toshio Fukuda, Zuankai Wang, and Yajing Shen. A bioinspired multilegged soft millirobot that functions in both dry and wet conditions. *Nature communications*, 9(1):1–7, 2018.

[13] Federico Ongaro, ChangKyu Yoon, Frank van den Brink, Momen Abayazid, Seung Hyun Oh, David H. Gracias, and Sarthak Misra. Control of untethered soft grippers for pick-and-place tasks. In *2016 6th IEEE International Conference on Biomedical Robotics and Biomechatronics (BioRob)*, pages 299–304, 2016.

[14] Phu Bao Nguyen, Byungjeon Kang, D. M. Bappy, Eunpyo Choi, Sukho Park, Seong Young Ko, Jong-Oh Park, and Chang-Sei Kim. Real-time microrobot posture recognition via biplane x-ray imaging system for external electromagnetic actuation. *International Journal of Computer Assisted Radiology and Surgery*, 13(11):1843–1852, Aug 2018.

[15] Gustaaf J Vrooijink, Alper Denasi, Jan G Grandjean, and Sarthak Misra. Model predictive control of a robotically actuated delivery sheath for beating heart compensation. *The International Journal of Robotics Research*, 36(2):193–209, 2017. PMID: 30814767.

[16] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105, 2012.

[17] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2015.

[18] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9, 2015.

[19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[20] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR, 2015.

[21] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 580–587, 2014.

[22] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[23] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, volume 1, pages 886–893. Ieee, 2005.

[24] Jonathan Huang, Vivek Rathod, Chen Sun, Menglong Zhu, Anoop Korattikara, Alireza Fathi, Ian Fischer, Zbigniew Wojna, Yang Song, Sergio Guadarrama, and Kevin Murphy. Speed/accuracy trade-offs for modern convolutional object detectors. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3296–3297, 2017.

[25] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[26] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(6):1137–1149, 2017.

[27] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[28] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.

[29] Jizhou Wang, Changhua Lu, and Weiwei Jiang. Simultaneous ship detection and orientation estimation in sar images based on attention module and angle regression. *Sensors*, 18(9), 2018.

[30] Xiabing Liu, Wei Liang, Yumeng Wang, Shuyang Li, and Mingtao Pei. 3d head pose estimation with convolutional neural network trained on synthetic images. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 1289–1293, 2016.

[31] Kota Hara, Raviteja Vemulapalli, and Rama Chellappa. Designing deep convolutional neural networks for continuous object orientation estimation. *arXiv preprint arXiv:1702.01499*, 2017.

[32] Ivo M. Baltruschat, Axel Saalbach, Mattias P. Heinrich, Hannes Nickisch, and Sascha Jockel. Orientation regression in hand radiographs: a transfer learning approach. In *Medical Imaging*, 2018.

[33] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[34] Chungkeun Lee, H. Jin Kim, and Kyeong Won Oh. Comparison of faster r-cnn models for object detection. In *2016 16th International Conference on Control, Automation and Systems (ICCAS)*, pages 107–110, 2016.

[35] Ayoub Benali Amjoud and Mustapha Amrouch. Convolutional neural networks backbones for object detection. In *International Conference on Image and Signal Processing*, pages 282–289. Springer, 2020.

[36] Miguel Juliá, Arturo Gil, and Oscar Reinoso. A comparison of path planning strategies for autonomous exploration and mapping of unknown environments. *Autonomous Robots*, 33(4):427–444, 2012.

[37] Stefano Scheggi and Sarthak Misra. An experimental comparison of path planning techniques applied to micro-sized magnetic agents. In *2016 International Conference on Manipulation, Automation and Robotics at Small Scales (MARSS)*, pages 1–6, 2016.

[38] Peter E. Hart, Nils J. Nilsson, and Bertram Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968.

[39] Edsger W Dijkstra et al. A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1):269–271, 1959.

[40] Seung-hyun Lim, Sun Woo Sohn, Hyoryong Lee, Donghyeon Choi, Eunsil Jang, Minhye Kim, Junhyeong Lee, and Sukho Park. Analysis and evaluation of path planning algorithms for autonomous

driving of electromagnetically actuated microrobot. *International Journal of Control, Automation and Systems*, 18(11):2943–2954, Jun 2020.

[41] Sven Koenig and Maxim Likhachev. Dˆ* lite. *Aaai/iaai*, 15, 2002.

[42] Sven Koenig, Maxim Likhachev, and David Furcy. Lifelong planning a*. *Artificial Intelligence*, 155(1):93–146, 2004.

[43] O. Khatib. Real-time obstacle avoidance for manipulators and mobile robots. In *Proceedings. 1985 IEEE International Conference on Robotics and Automation*, volume 2, pages 500–505, 1985.

[44] L.E. Kavraki, P. Svestka, J.-C. Latombe, and M.H. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, 12(4):566–580, 1996.

[45] Steven M LaValle et al. Rapidly-exploring random trees: A new tool for path planning. 1998.

[46] Esmaeel Khanmirza, Morteza Haghbeigi, Milad Nazarahari, and Samira Doostie. A comparative study of deterministic and probabilistic mobile robot path planning algorithms. In *2017 5th RSI International Conference on Robotics and Mechatronics (ICRoM)*, pages 534–539, 2017.

[47] Gustaaf J. Vrooijink, Momen Abayazid, Sachin Patil, Ron Alterovitz, and Sarthak Misra. Needle path planning and steering in a three-dimensional non-static environment using two-dimensional ultrasound images. *The International Journal of Robotics Research*, 33(10):1361–1374, 2014. PMID: 26279600.

[48] Richard Szeliski. *Computer vision: algorithms and applications*. Springer Science & Business Media, 2010.

[49] John Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(6):679–698, 1986.

[50] Martin A. Fischler and Robert C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, jun 1981.

[51] Raúl Rojas. *Neural networks: a systematic introduction*. Springer Science & Business Media, 2013.

[52] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[53] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

[54] Conv2d - pytorch 1.10.1 documentation. `https://pytorch.org/docs/stable/generated/torch.nn.Conv2d.html#torch.nn.Conv2d`.

[55] Wikimedia Commons typical cnn. `https://commons.wikimedia.org/wiki/File:Typical_cnn.png`. Accessed: 2022-01-07.

[56] Jakub Sikorski, Imro Dawson, Alper Denasi, Edsko E.G. Hekman, and Sarthak Misra. Introducing bigmag — a novel system for 3d magnetic actuation of flexible surgical manipulators. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3594–3599, 2017.

[57] Jakub Sikorski, Alper Denasi, Giuseppe Bucchi, Stefano Scheggi, and Sarthak Misra. Vision-based 3-d control of magnetically actuated catheter using bigmag—an array of mobile electromagnetic coils. *IEEE/ASME Transactions on Mechatronics*, 24(2):505–516, 2019.

[58] Martin Golubitsky, Ian Stewart, Pietro-Luciano Buono, and J. J. Collins. Symmetry in locomotor central pattern generators and animal gaits. *Nature*, 401(6754):693–695, Oct 1999.

[59] Venkatasubramanian Kalpathy Venkiteswaran, Luis Fernando Peña Samaniego, Jakub Sikorski, and Sarthak Misra. Bio-inspired terrestrial motion of magnetic soft millirobots. *IEEE Robotics and Automation Letters*, 4(2):1753–1759, 2019.

[60] Clariusdev/cast - release version 8.6.0. `https://github.com/clariusdev/cast/releases/tag/8.6.0`. Accessed: 2021-07-08.

[61] Qt creator: documentation. `https://doc.qt.io/qt.html`. Accessed: 2021-03-03.

[62] Opencv documentation. `https://docs.opencv.org/4.x/`. Accessed: 2021-03-03.

[63] Boost: C++ library. `https://www.boost.org/`. Accessed: 2021-09-08.

[64] Satoshi Suzuki and KeiichiA be. Topological structural analysis of digitized binary images by border following. *Computer Vision, Graphics, and Image Processing*, 30(1):32–46, 1985.

[65] S. Brlek, G. Labelle, and A. Lacasse. The discrete green theorem and some applications in discrete geometry. *Theoretical Computer Science*, 346(2-3):200–225, Nov 2005.

[66] AW Fitzgibbon and RB Fisher. A buyer's guide to conic fitting. *Procedings of the British Machine Vision Conference 1995*, 1995.

[67] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2 edition, 2004.

[68] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[69] Simon J Julier and Jeffrey K Uhlmann. New extension of the kalman filter to nonlinear systems. In *Signal processing, sensor fusion, and target recognition VI*, volume 3068, pages 182–193. International Society for Optics and Photonics, 1997.