BEN-GURION UNIVERSITY OF THE NEGEV

THE FACULTY OF ENGINEERING SCIENCES

DEPARTMENT OF MECHANICAL ENGINEERING

Design, Experiments and Implementation of Reinforcement Learning in RSTAR Robot for Search and Rescue Applications

THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE M.Sc. DEGREE

By: Liran Yehezkel

Supervised by: Dr. David Zarrouk

May 2020

May 2020

BEN-GURION UNIVERSITY OF THE NEGEV

THE FACULTY OF ENGINEERING SCIENCES

DEPARTMENT OF MECHANICAL ENGINEERING

Design, Experiments and Implementation of Reinforcement Learning in RSTAR Robot for Search and Rescue Applications

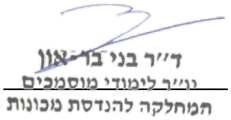
THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE M.Sc. DEGREE

By: Liran Yehezkel

Supervised by: Dr. David Zarrouk

Author: Liran Yehezkel  Date: 31.5.2020

Supervisor: David Zarrouk  Date: 31.5.2020

Chairman of graduate studies committee:  Date: 15.7.20

May 2020

Abstract

This thesis presents the Rising STAR (RSTAR) a newly developed crawling robot capable of reconfiguring its shape and moving the position of its center of mass. RSTAR belongs to the family of the STAR robots with similar sprawling capabilities allowing it to run in a planar configuration, either upright or inverted and change its mechanics from the lateral to the sagittal planes. The RSTAR is also fitted with four bar extension mechanism (FBEM) allowing it to extend the distance between its body and legs.

This combination of sprawling and extension mechanisms enables RSTAR to overcome extremely challenging obstacles, crawl over flexible and slippery surfaces and even climb vertically in a tube or between two walls. The robot can extend its height and width three-fold and move its center of mass both in the fore-aft and vertical directions.

We first describe a kinematic model and the dynamical analysis conducted to improve the design of the robot and evaluate its strength and motor requirements. Based on this analysis, we designed and built a 3D printed prototype and experimentally tested it. The robot can run upside down and climb over obstacles that are even higher than the diameter of its wheeled legs using a turtle-like gait. To increase its mobility, RSTAR can be fitted with wheels or spoked legs or a combination of the two, giving it superior ability to engage different terrains.

However, defining trajectories that utilize the robot's capabilities is difficult, especially when complex maneuvers are required. Here, we show how the use of reinforcement learning can serve to determine optimal strategies to overcome three typical obstacles: squeezing through two adjacent obstacles, ducking underneath an obstacle and climbing over an obstacle. We detail the implementation of the Q learning algorithm in a simulation environment with a physical engine (UNITYTM) for learning a feasible path in a minimum number of steps.

We compare the trajectory found by the algorithm to trajectories devised by six human experts for the RSTAR simulation. Our results show that the algorithm was able to find a feasible trajectory in all cases. Moreover, the trajectories found by the algorithm were shorter than the trajectories determined by the human experts. Finally, we present experiments where the physical RSTAR robot can overcome different obstacles using the trajectories found in the simulation by the Q Learning algorithm (see attached videos [24][48]).

Based on this research we published two papers:

1. "Rising STAR: A Highly Reconfigurable Sprawl Tuned Robot," in IEEE Robotics and Automation Letters.
2. "Beyond Human Performance: Overcoming Obstacles with a Reconfigurable Robot Using Reinforcement Learning" submitted for IEEE Access.

Key Words

Robotics; Mechanical design; Crawling Robot; Reconfigurable Robot; Sprawl Tuning; Reinforcement Learning; Q Learning.

Acknowledgments

First, I would like to express my gratitude to my supervisor, Dr. David Zarrouk, for the guidance and encouragement during my research and thesis writing. I would like to thanks him for all the support, patient advice and the trust he placed in me.

In addition, I would like to thank my family for loving and supporting me and for providing me with continuous encouragement throughout my years of study and throughout the research process and the thesis writing.

Table of Contents

Nomenclature

Symbol	Units	Meaning
θ_s	degrees	The sprawl angle of the RSTAR
θ_F	degrees	The FBEM angle of the RSTAR
$\theta_{F-\max}$	degrees	The maximum possible FBEM angle
Φ	degrees	The maximum tilt angle that the robot can statically withstand before tipping over
m_{body}	grams	The mass of the RSTAR's body
m_{leg}	grams	The mass of the RSTAR's legs
ΔCOM_{height}	mm	Possible movement of COM in the fore-aft direction
ΔCOM_{height}	mm	Possible movement of COM in the vertical direction
$\Delta foreaf$	mm	Possible movement of the RSTAR's legs relative to its body in the fore-aft direction
$\Delta height$	mm	Possible movement of the RSTAR's legs relative to its body in the vertical direction
L_{bar}	mm	The length of legs' bars
L_{leg}	mm	The radius of the wheels
L_1	mm	Typical length on the robot design
L_2	mm	Typical length on the robot design

L_3	mm	Typical length on the robot design
L_w	mm	Typical length on the robot design
g	m/s^2	Gravity
F_{normal}	N	The forces acting on one side of the legs in the normal direction
F_{side}	N	The forces acting on one side of the legs in the side direction
$F_{foreaft}$	N	The forces acting on one side of the legs in the fore-aft direction
T_{sprawl}	Nmm	The torque of the sprawl mechanism
T_{FBEM}	Nmm	The torque of the FBEM
T_{leg}	Nmm	The torque of the legs
μ	[ ]	Coefficient of friction
α	[ ]	Learning rate
γ	[ ]	Discount factor

List of Tables

List of Figures

1 Introduction

Miniature search and rescue ground robots that can be used in disaster areas have numerous advantages. There are several examples of palm sized search and rescue ground robots that can crawl, run and climb over obstacles. These include the Mini-Whegs[1], Dyna-RoACH [2], DASH [3], iSprawl [4], OctoRoACH [5], RHex [6], Sprawlita [7] , STAR [8], 1STAR [9] and TAYLRoACH [10]. They can be deployed in large numbers to scout large areas while keeping operators out of harm's way. Their low weight and small size are important for efficiently performing many tasks.

Designing minimally actuated mechanisms is imperative at this scale given the inherent difficulty of implementing controlled active leg joints. Passive mechanical elements such as using springy legs and damping systems to achieve high speed while maintaining stability such as found in insects have been investigated by multiple research groups [11][12]. They have developed crawling models such as the spring loaded inverted pendulum model (SLIP) [13]-[15], which describes the locomotion of insects in the sagittal plane and in-plane (lateral) models of locomotion [16]-[19]. In parallel, multiple attempts have been made to produce robots with reconfigurable kinematics to overcome obstacles [20]-[22].

In a previous work [8], we presented a sprawl tuned autonomous robot (STAR) which can actively adjust its sprawl angle to transform its dynamics between the lateral and the sagittal planes through the use of a variable sprawl angle. STAR exhibited many unique capabilities such as moving on varying terrain surfaces and traversing obstacles. The RSTAR robot presented in this thesis (Figure 1.1) belongs to the family of STAR robots [8][9][23]; i.e., it can also vary its sprawl angle. However, thanks to its four bar extension mechanism (FBEM), this design has superior capabilities compared to our previously designed STAR in overcoming obstacles by reconfiguring its mechanics, and moving its center of mass (COM).

At low sprawl angles, the contact angle of the foot with the surface is reduced, which minimizes collisions and reduces the uncontrolled vertical dynamics, resulting in smooth operation of the robot at all speeds. The sprawl angle can also be used to change the width and height of the robot so it can squeeze itself in between or under obstacles and ride over them. The characteristic length of the robot is 15 cm and has a weight of 308 grams including the battery and control board for either human control or autonomous operation. To improve stability and energy consumption, the robot is fitted with radially spoked legs or wheels or both (on each side).

Figure 1.1 - The Rising STAR (RSTAR) robot is a highly reconfigurable robot for search and rescue proposes (see video [24]). The sprawl mechanism allows the robot to move in and out of the plane whereas the four bar extension mechanism (FBEM) extends the arms holding the legs and relocates the center of mass.

Although non-orthodox mechanisms can provide advanced solutions to various maneuverability needs, nevertheless executing trajectories involving these capabilities remains a complex problem. Here, we also describe a way to overcome this problem based on autonomous learning. Teleoperating a reconfigurable robot like the RSTAR requires a highly trained operator since different mechanisms often need simultaneous actuation.

It is preferable to reduce human intervention by facilitating autonomous robot operations. One way to increase robot autonomy is to teach the robot to autonomously perform common operations. In the case of off-road missions these include climbing over obstacles, and maneuvers around or underneath obstacles. Machine learning, and specifically reinforcement learning (RL), can be used to teach the robot how to perform such maneuvers offline by repetitively performing experiments in a physical or simulated environment.

Considerable efforts have been devoted over the years to devising methods for the autonomous learning of robotic motion trajectories. One of the most popular is reinforcement learning (RL)[27][29]. However, most researchers have concentrated on devising ways for the robot to learn how to avoid obstacles [30][34]. For example, Lee et al. [30] applied RL to navigate between obstacles with a quadruped robot. Similarly, Zhang et al. [31] use deep RL to address navigation issues faced in cluttered environments on previously unknown rough terrain.

Overcoming an obstacle (e.g., climbing over, or ducking under) rather than avoiding it, can lead to more efficient operations, and in some cases be critical for mission success. Successful operation typically requires a combination of kinematic and dynamic maneuvers. The motion plan must provide a feasible path as a function of the robot's dynamic capabilities. The autonomous learning of such trajectories requires taking complex dynamic considerations into

account. It must also be done in setups that preserve the safety of the robot during the learning process. To date, very few projects have successfully met this challenge. Notable exceptions include M. Totani et al. [35] who used RL to learn a step climbing method for a crawler-type rescue robot. Dutta et al. [36] used multi-agent learning for locomotion learning in modular self-reconfigurable robots (MSRs).

One of the main drawbacks to RL is that learning times may be prohibitively long, when problem dimensionality is high. In robotic systems problem dimensionality grows with the number of controlled dimensions (e.g., actuators). Due to the need for multiple learning episodes and due to potential hazards to the robotic system during leaning epochs, many researchers opt to use simulation platforms for the leaning process. A major challenge in such cases is reliably ensuring the transferability of the learning from the simulation to the physical environment.

Many systems that apply RL have a very high degree of interaction between their multiple degrees of mobility and the environment, e.g., multi-rotor drones [37], where complex aerodynamic effects are caused by interactions between multi-rotor airflow and the environment. Other systems require intricate parameter tuning, e.g., underactuated legged robots [38][39] that are required to perform highly dynamic motion involving balance. In such cases the feasible solutions require very accurate determination. This necessitates searching for solution in continuous action spaces and this strains the transferability of simulation results. Much effort has been devoted in recent years for developing efficient RL algorithms suitable for such continuous action environments, that are not sensitive to hyperparameters, that require a manageable amount of leaning sequences, and for which results are transferable from simulated to physical environments [37]-[40].

The unique design of the RSTAR robot affords very high maneuverability with a relatively small number of actuators (4). This makes the solution search space manageable. Moreover, the inherent stability of the RSTAR and its decoupled degrees of freedom (the different mechanisms and wheels can be actuated almost independently), facilitates relatively large tolerances to the feasible control policies. This facilitates using the much simpler discrete RL methods. Moreover, it increases the robustness of the solution to small discrepancies in the modeled dynamics, and therefore, increases the transferability of the solutions to the physical world. Still, as motion dynamics are key to mission success, a physical simulation engine is required for meaningful learning. In addition to simplifying the learning process, discretizing the action space simplifies

the required run-time control, as the polices can be represented as discrete sequences of required actions.

In this thesis, we focus our research on applying RL for autonomously learning how to exploit RSTAR's advanced reconfiguration capabilities for overcoming obstacles by climbing over ducking underneath and squeezing through. We apply the Q-learning algorithm, an off-policy, temporal differencing model free RL algorithm [29]. A model free algorithm was selected because the formulation of the dynamic equations of RSTAR is complex and prone to error which may can impede learning. We opted to use the algorithm with a discrete state and action space rather than applying deep RL methods (e.g., Deep-Q). Since motion dynamics is key to mission success, learning must be done in an environment reminiscent to the physical environment to be meaningful. Therefore, at the very least, a physical simulation engine is required. In such environments learning epochs are indeed shorter than in physical environments, but still relatively time consuming and thus limited. This makes the application of deep-RL methods costly.

This thesis is divided into two main parts: a mechanical part and a learning part. In the first part (chapters 3-5), the mechanical design of the RSTAR is presented in Chapter 3. The kinematics and dynamics of the robot are described in Chapter 4 and the robot's capabilities are depicted in series of experiments in Chapter 5. In the second part (chapters 6-8) the virtual environment used for the learning process and the learning algorithm itself are explained in detail (Chapter 6). The results of the learning process are presented in Chapter 7 and in Chapter 8 the results are implemented on the real RSTAR in set of experiments.

2 Background

This section reviews some theoretical background relevant for this research including review about search and rescue robots and explanation about Q learning algorithm.

2.1 Reconfigure Abilities in The Use of Search and Rescue Robots

Miniature crawling robots have been developed for off- road tasks such as search and rescue. Their small size, low weight and high navigability enable their deployment in large numbers to quickly inspect a large area. At the same time, however, small mobile robots are limited in their ability to traverse on varying terrains and climb over obstacles because the objects around them become relatively larger.

This brief overview provides examples from the literature on search and rescue robots developed to adapt to overcome obstacles. Some of these robots can alternate between leg and wheel modes to take advantage of both systems whereas others are designed with reconfigurable kinematics to overcome obstacles.

Wheel Transformer, a wheel-leg hybrid robot [20], utilizes a transformable wheel that combines the advantages of both circular and legged wheels. When it encounters an obstacle, the wheel transforms into a legged wheel with three legs (Figure 2.1) and can climb over steps higher than the wheel's radius. The switch from the circular to the legged configuration implements the frictional force between the wheel and obstacle without additional actuators.

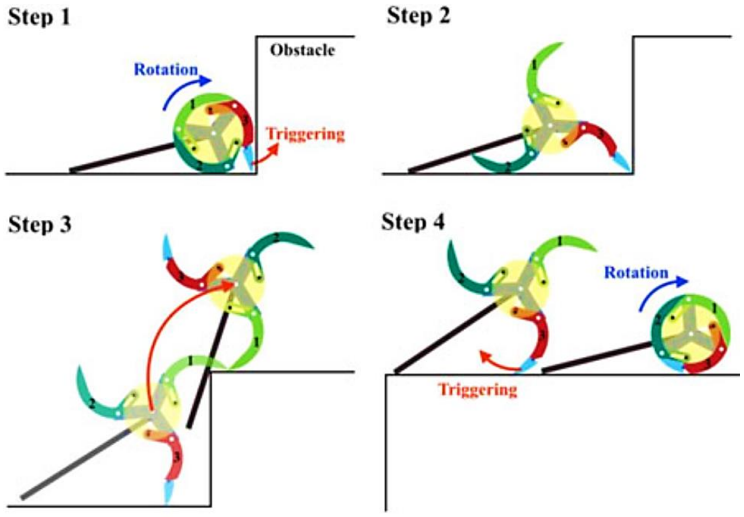


Figure 2.1 - a wheel-leg hybrid robot, combines circular and legged wheels.

RiSE (Robots in Scansorial Environments) [21] is a legged robot capable of locomotion on both the ground and a variety of vertical building surfaces (Figure 2.2.a) including brick, stucco, and crushed stone. It climbs quietly without suction, magnets, or adhesives. RiSE has a range of motion of 190 degrees in its "wing" degrees of freedom (Figure 2.2.b), which allows the robot to walk with its legs underneath its body and climb with its legs in a sprawl position.

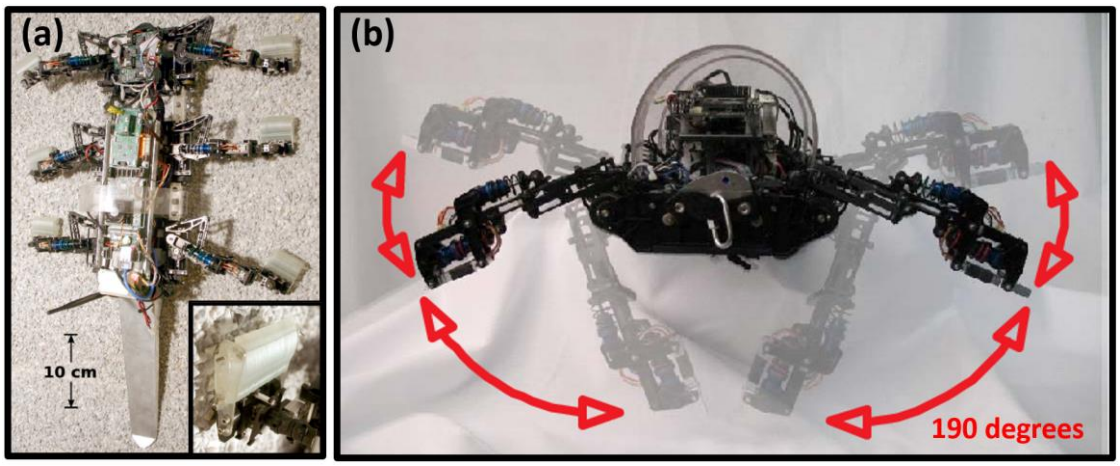


Figure 2.2 – The RiSE robot, adjusting the angle between it body and legs it is capable to crawl on the ground and on a variety of vertical building surfaces.

PUFFER (Pop-Up Flat Folding Explorer Robot) [22] is a palm-sized, origami-inspired wheeled rover designed to accompany larger spacecraft on future missions, and serve as a mobility enhancement to provide access to new terrains. PUFFER rovers are constructed with a collapsible "pop-up" chassis that folds into a compact volume for storage, as shown in Figure 2.3. PUFFER's small size and folding chassis also provide mobility benefits that enable PUFFER to maneuver in extreme terrains inaccessible to the parent. This partial collapsibility can also be used to lower the platform's center of gravity when climbing steep inclines.

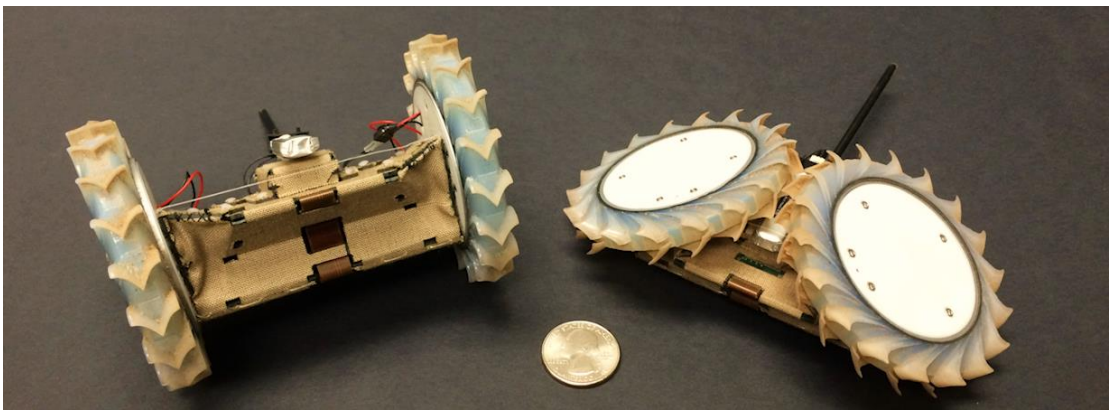


Figure 2.3 - PUFFER prototype, expanded (left) and folded (right) configurations.

In a previous work we presented the STAR (Sprawl Tuned Autonomous Robot) [8],[10] which is the original version of the RSTAR. The STAR (Figure 2.4.a) possesses high-mobility capabilities combining the benefits of wheeled and legged locomotion. It can actively adjust its sprawl angle (Figure 2.4.b-d) to transform its dynamics between the lateral and the sagittal planes through the use of a variable sprawl angle. In particular, large sprawl angles were found to be efficient on uneven terrain, whereas the low sprawl posture is better suited for traveling over smooth surfaces. The sprawl angle can also be used to change the width and height of the robot so it can squeeze itself in between or under obstacles and ride over them. STAR exhibited many unique capabilities such as moving on varying terrain surfaces and traversing obstacles.

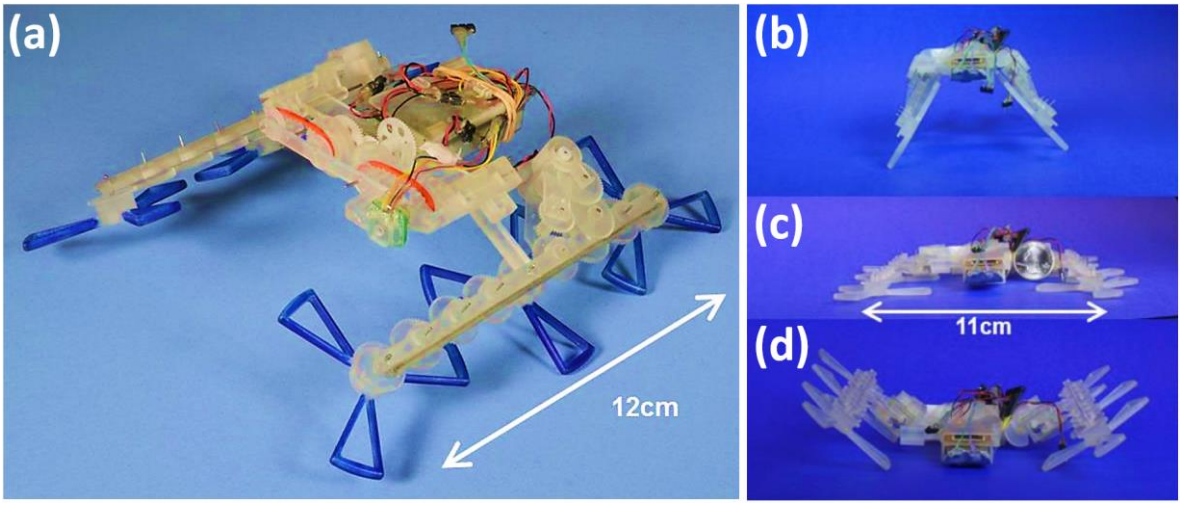


Figure 2.4 – The STAR robot, the original version of the RSTAR robot. It can change its sprawl angle of its legs from nearly flat posture to vertically oriented legs.

The RSTAR can also change its sprawl angle. However, by adding its four bar extension mechanism (FBEM), it has modified capabilities with respect to overcoming obstacles compared to the original design of the STAR. The design of the RSTAR will be presented in more details in Chapter 3.

2.2 Q Learning Algorithm

This section reviews the Q learning algorithm, we used for learning optimal strategies to overcome obstacles. For a better understanding of the Q learning algorithm, this overview also includes an example of the algorithm's implementation on a simple cube (Figure 2.5). The goal is to get the cube to the target by implementing the least number of actions and to do so while complying with certain rules of physical behavior; i.e., the cube can only move upwards when it is positioned in front of the step.

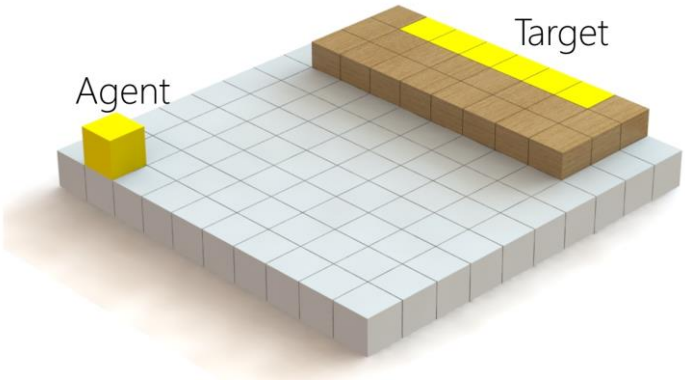


Figure 2.5 – Cube learning to navigate in space toward the target using a Q learning algorithm.

2.2.1 Basic Principles of Q Learning Algorithm

Q learning is a form of model-free reinforcement learning (RL), which is a type of machine learning. RL algorithms [41] are based on a Markov Decision Process (MDP), in which the probability to reach a given state depends solely on the previous state and the action that was taken. Throughout the learning process the algorithm learns a policy, π, that maps states to actions.

In the Q-learning algorithm [42], an action-value function $Q^{\pi}(s, a)$, also called Q function, is estimated over the learning process and stored in a tabular representation. Each state-action pair has a "Q value" that represents the expected sum of rewards the agent expects to receive by executing action, a, from state, s, following policy π. The goal of the algorithm is to learn the optimal policy that maximizes the total expected rewards.

The algorithm is based on repeating iterations of learning and experience gathered by receiving reward feedback for taking actions. The Q-learning algorithm is guaranteed to converge to the optimal Q function under certain conditions [43]. After it converges, the optimal Q function can be used to determine the optimal action to take in a given state.

The main components of Q learning algorithm (see Figure 2.6) are:

- The algorithm itself, also called the agent.
- The environment of the problem, divided into range of possible states.
- Actions the agent can take when interacting with the environment.
- The reward function, which defines the reward for taking action a from state s.
- The Q function that represents the expected sum of rewards for each action-state pair.

The learning process is iterative, such that at each iteration the agent takes the following steps:

- The agent observes its current state, s_n, in the environment.
- The agent selects and performs an action, a_n.
- The subsequent state, s_{n+1}, is observed.
- Reward, r_n, is received from the reward policy.
- The algorithm updates the Q value of the state-action pair using the received reward.

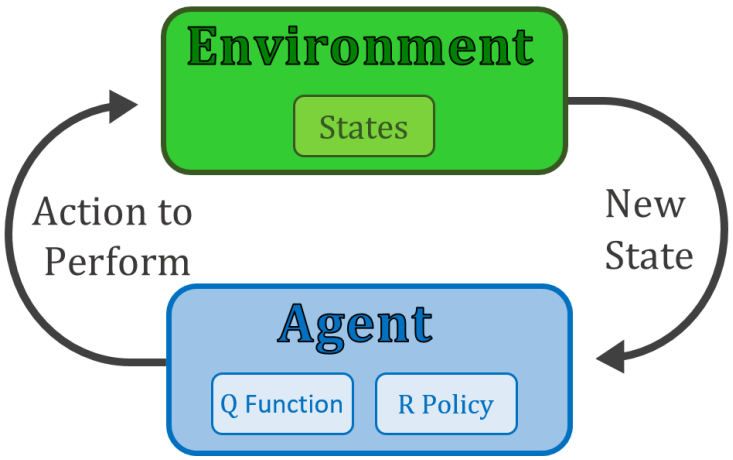


Figure 2.6 – The interaction of the main components in the Q Learning algorithm.

As long as the number of iterations increases, performance improves. By repeating enough iterations will allow the agent to determine the optimal policy for the full range of possible states.

2.2.2 States Determination

The environment is divided into discrete states, where each state can be a composite of multiple components that define the agent in the environment such as position, orientation, etc. The discretization of the environment constitutes a tradeoff between accuracy and the complexity of the learning process. On one hand, dividing the environment into very small increments will

improve the accuracy of the definitions for the agent with respect to its surroundings, but on the other will increase the number of states the algorithm has to learn and hence increase the learning time.

In the cube example (Figure 2.5), the state is determined to have two position components: an altitude position, $[y]$, and a position on the 2D plane, $[p]$. According to the state determination, the discretization of the environment takes place as follows: the altitude dimension is divided into two possible heights: floor height ($y = 1$) and step height ($y = 2$), and each of the two planes is divided into (nxm) squares as illustrated in Figure 2.7. This division enables the agent's state, s, to be defined by two parameters: altitude, y, and position, p, so: $s = [y][p]$.

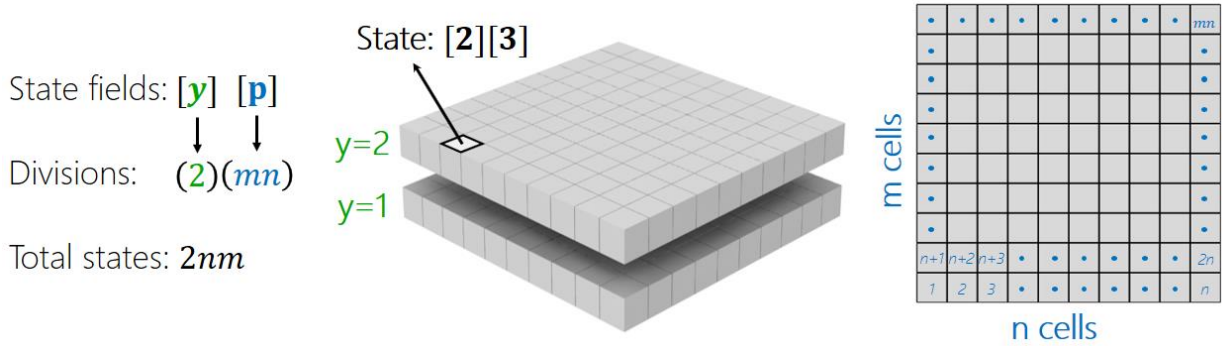


Figure 2.7 – State determination for the cube example, where the state is defined by the plane altitude and by its position on the plane. Accordingly, the environment is divided into discrete states that define the location of the cube on it.

2.2.3 Action Possibilities

Actions transfer the agent from state to state. Actions are associated with the actions the agent can perform to reach the target. In the cube example, the cube can move in five possible directions: forward, backward, left, right and upwards.

2.2.4 Q Function

The Q function contains values for each action-state pair. Each value $Q(a, s)$ estimates the sum of rewards the agent is expected to receive for taking action, a, from state, s. In cases of discrete problems, the Q function can be presented as a Q matrix, where the number of existing matrix values is a multiplication of the number of actions by the number of states.

The Initialization of the Q matrix influences the learning time of the algorithm [44] considerably and affects the learning policy at the beginning of the learning process. By setting

even initial conditions, all states are rated equally and the algorithm has to learn all the possible states to converge to a solution. Uneven initial conditions can shorten the learning time by distinguishing between attractive states and states that make no contribution to reaching the target. Attractive states are initialized with high values relative the other states.

In the cube example, the Q matrix is a 3D matrix (action dimensions and a 2D state dimension), so that each Q value is defined by: Q = [a][y][p] (see Figure 2.8). Each state s = [y][p], is associated with five different values, where each value is related to another action taken from state s.

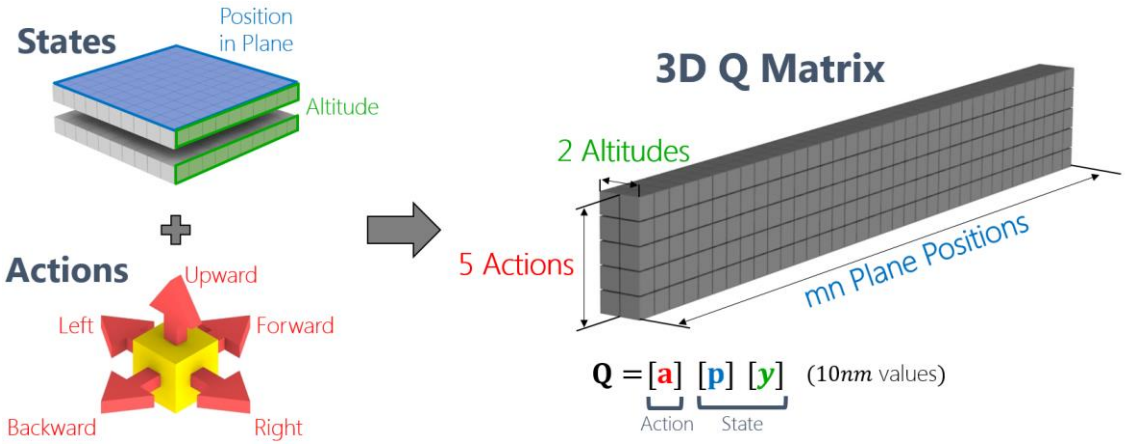


Figure 2.8 –The Q matrix for the cube example, where each cell in the matrix contains the expected sum of rewards for taking action, a, from plane position, p, at altitude y.

2.2.5 Exploitation policy vs. Exploration policy

When selecting an action to perform, the agent needs to resolve the exploitation-exploration dilemma. In case of exploitation, the agent exploits its current knowledge and selects the action with the highest Q value (greedy action). In the case of exploration, it randomly selects one non-greedy action and improves its estimate of the non-greedy action values.

Selecting actions solely by exploitation usually results in finding and preferring local optima rather than the global goal. Exploration avoids this problem by exploring new states and actions even though the agent is more likely to continue to randomly explore areas that are not of interest and can increase the learning time. Therefore, the policy adopted while learning must be chosen to balance these concerns and promote efficient convergence of the algorithm.

One possible way to combine exploitation and exploration is by using the ϵ-greedy policy. In this policy the agent starts exploring from the outset with high probability of taking random actions, such that as the learning process proceeds, the probability for exploration decreases and

actions are more likely to be selected by exploitation (greedy policy). Given that ϵ represents the probability for taking random actions we have:

$$\begin{aligned} &\text{if}\left(\text{probability} > \epsilon\right) \\ &\quad a_n = \max_{a'} Q\left(a', s_n\right) \\ &\text{else} \\ &\quad a_n = \text{random}\left(a_i\right) \\ &\quad \text{decrease} \quad \epsilon \end{aligned} \tag{1}$$

2.2.6 Reward Function

The reward function determines the immediate reward after executing an action from a given state. The reward function plays an important role in RL algorithms [44], since a reward function tailored to the problem can accelerate the convergence rate whereas inappropriate rewards can increase the learning time and even cause the algorithm to diverge.

For this reason, several methods have been proposed for designing the reward function. In [45] a methodology for designing reward functions was suggested that takes advantage of implicit domain knowledge. [46] explored reward shaping, where the rewards from the environment are augmented with additional rewards.

For the cube example, the reward function is illustrated in Figure 2.9. Because the algorithm seeks a solution with the maximum reward (greedy policy), most actions are rewarded negatively to ensure a solution with minimum number of actions. To reach a solution that involves the right physical behavior, nonphysical actions are rewarded more negatively than other actions. For example, the punishment for movement on the upper plane (when the step is not located underneath) is twice the punishment for navigating on the floor.

To accelerate the convergence, positive rewards are also delivered for taking specific actions from specific states (advancing forward in certain states and climbing from states located in front of the step). After getting to the target, a high positive reward is delivered to encourage the agent to return to the target state in the next iterations.

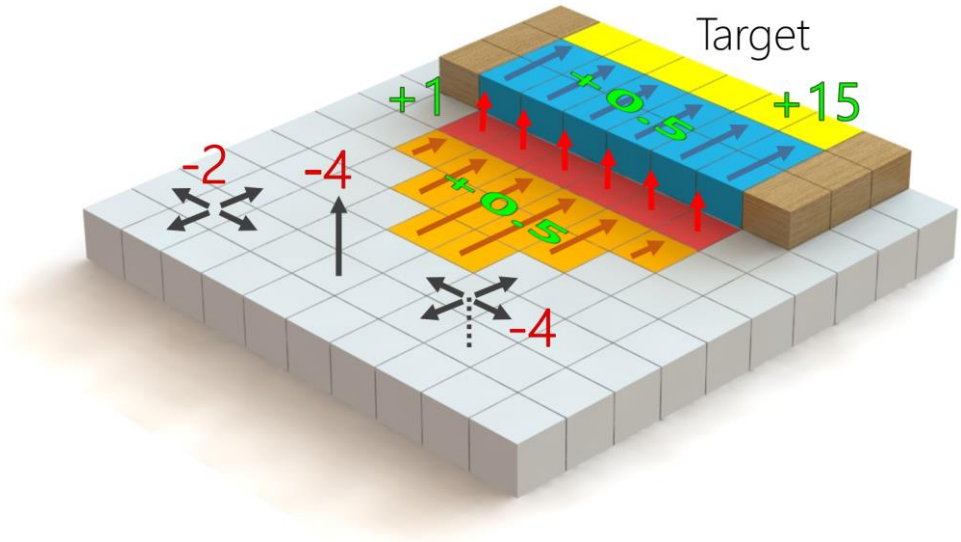


Figure 2.9 - The reward function for the example of a cube navigating towards the defined target.

2.2.7 Q Function Update

The update of the matrix Q is executed using:

$$Q(a_n, s_n) \leftarrow (1-\alpha)Q(a_n, s_n) + \alpha[r_n + \gamma \max_{a'} Q(a', s_{n+1})]. \quad (2)$$

where r_n is the reward for taking action a_n at state s_n, s_{n+1} is the state of the agent after taking action a_n. Action a' is the one associated with the highest possible Q(s_{n+1}, a) value. The learning rate α is set to between 0 and 1. Setting α to 0 means that the Q-values are never updated; hence, nothing is learned. Setting a high value such as 0.9 means that learning can occur quickly. The discount factor γ is also set to between 0 and 1. This models how important future rewards are to the current state. Mathematically, the discount factor needs to be set to less than 1 for the algorithm to converge.

To provide insights into equation (2) we present an update stage on the cube problem. The cube is positioned in state $S_n = [35][1]$ and moves right (let's assume this action is presented with index 4 on the Q matrix). For a learning rate of 0.6, a discount factor of 0.9 and a given reward of -2 (see Figure 2.9), the update of the Q value will be in the form of:

$$Q(4,35,1) \leftarrow (1-0.6)\cdot Q(4,35,1) + 0.6\cdot[-2 + 0.9 \max_{a'} Q(a,36,1)]. \quad (3)$$

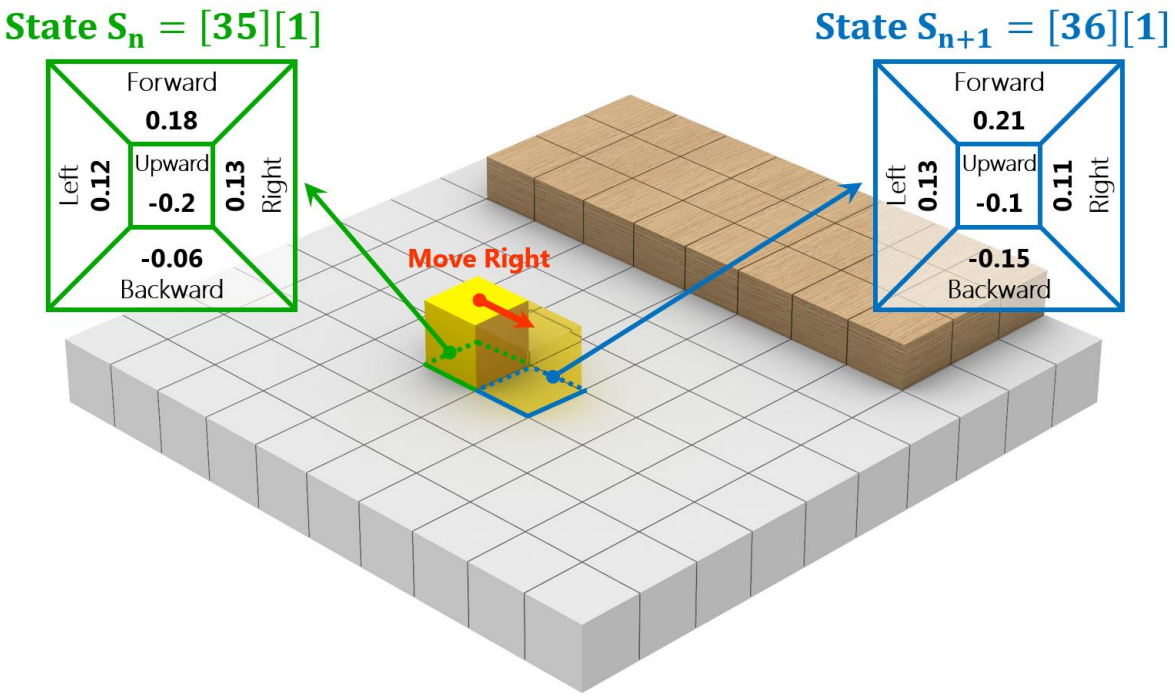


Figure 2.10 - Example of a Q update in the cube problem. The cube moves right from state S_n and the related Q value is updated.

Note that the state $S_n = [35][1]$ is related to five different Q values (one value for one action), and the updated Q value is the one associated with the action made by the cube, i.e., moving right (with the index 4).

To take future rewards into account, the algorithm seeks the maximum Q value in the subsequent state S_{n+1} (this value is associated with action a'). In the case above, the maximum Q value of state S_{n+1} is related to $a' = 1$ (the index for moving forward). By assigning the appropriate values from Figure 2.10, the updated Q value for moving right from state $S_n = [35][1]$:

$$\begin{aligned} Q(4,35,1) &\leftarrow 0.4 \cdot Q(4,35,1) + 0.6 \cdot [-2 + 0.9 \max_{a'} Q(1,36,1)] = \\ &= \quad 0.4 \cdot 0.13 + 0.6 \cdot [-2 + 0.90.21] = -1.035 \end{aligned} \quad . \qquad (4)$$

3 Mechanical Design and Manufacturing

This section specifies the mechanical design of the RSTAR and its main parameters, as well as the materials used for manufacturing of the robot. The primary design goal of RSTAR is to achieve a highly maneuverable robot capable of crawling over different terrains and overcoming obstacles. The robot must also be lightweight and capable of carrying the substantial payloads that may be required to perform search and rescue missions including cameras, communication equipment and sensors. One of the key requirements was to keep the cost of transport (COT) of the robot as low as possible. These goals are achieved by the combination of the active sprawl angle together with the FBEM, which allow the robot to transform its kinematics and substantially change its dimensions to overcome obstacles.

3.1 Robot Design

The RSTAR consists from a main rigid body and a pair of legs fitted with wheels and spoke legs as presented in Figure 3.1. The body holds the controllers, the onboard batteries, the sprawl mechanism and FBEM mechanism. Each of these mechanisms is actuated by a single motor. Both sides of the robot are phased together and move symmetrically relative to its center. The wheels of each legs are actuated by a single motor. In total, the robot has four motors.

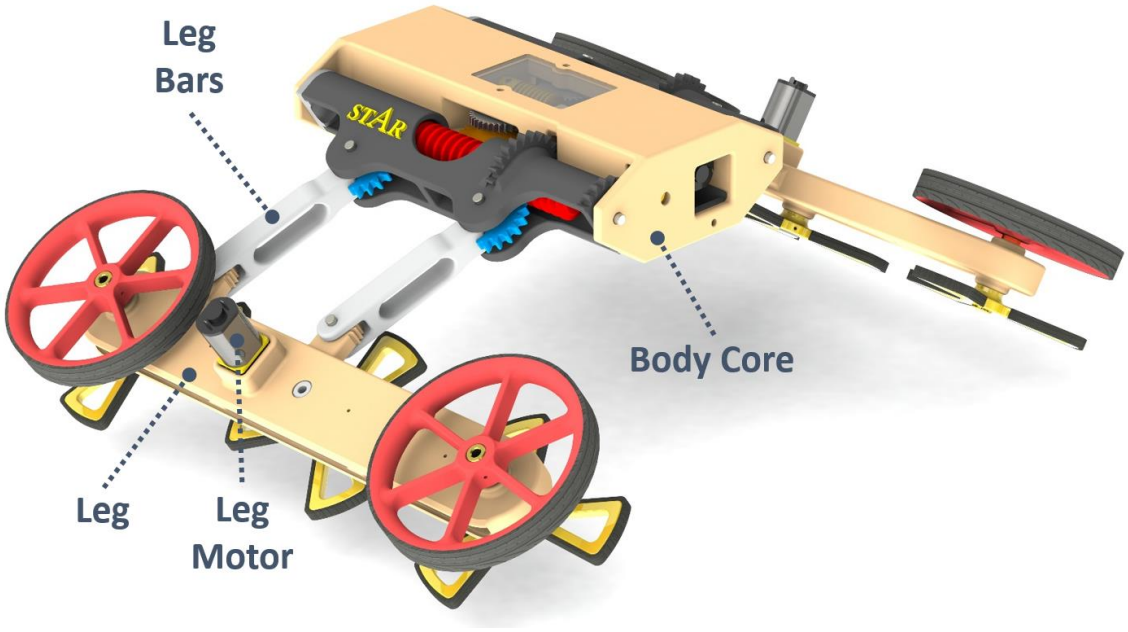


Figure 3.1: Isometric view of the RSTAR robot.

3.1.1 The Sprawling mechanism

The relative angle between the legs and the main body, as presented in Figure 3.2, forms the sprawl angle θ_s , which is defined as $\theta_s = 0$ when the legs are coplanar with the ground. The sprawl angle can be varied in the range [-90, 90] (the positive sense of the sprawl angle is downwards), as shown in Figure 3.2, allowing the robot to continue running in the same direction even when upside down.

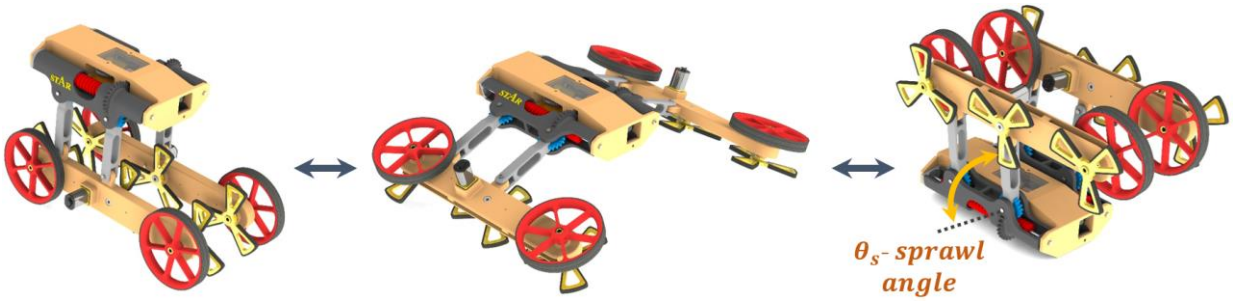


Figure 3.2 – Definition of the sprawl angle, the relative angle between the legs and the main body. Changing the sprawl angle can increase or lower the robot width and height.

The sprawl angles at both sides of the robot are actuated symmetrically through a single motor and four spur gears which provides 16:25 gear ratio to increase the motor torque, see Figure 3.3. This mechanism ensures both sides to rotate at identical sprawl angle but in opposite directions. As depicted in Figure 3.3, two extra spur gears and an angular potentiometer are also attached to one of the robot's legs to measure the sprawl angle.

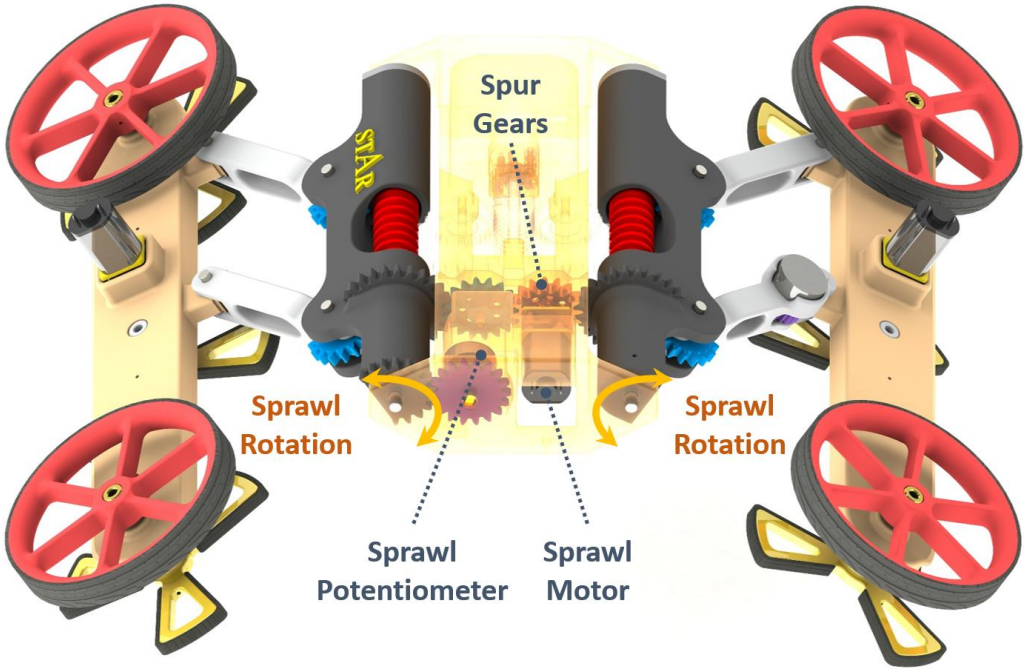


Figure 3.3 - The mechanical design of the sprawl rotation mechanism, consists from DC motor and four spur gears that ensure symmetrically rotation of both sides relative to the main body.

3.1.2 The Four Bar Extension Mechanism (FBEM)

The rotation angle of the four bar extension mechanism is denoted by θ_F and presented in Figure 3.4. The FBEM angle can be varied in the range [-72, 72], the rotation angle θ_F is defined zero when the two bars are perpendicular to the body and the legs.

Figure 3.4 - Definition of the FBEM angle. By changing the FBEM angle the width and the length of the robot can be changed.

The FBEM is attached to the sprawl mechanism and rotates together with it because of the shape of the rack spur gear (see Figure 3.5). Both sides of the FBEM are actuated using the same motor and are symmetric relative to the body. Worm gear is used to provide a high gear ratio and self-locking when not activated. Rotating the gears attached to the worm gear results in linear movement of the rack spur gear along the shaft. The movement of the rack spur gear rotates the two parallel bars of the FBEM. That bars that connect the sprawl mechanism to the robot legs and are synchronized to rotate at the same speed. An angular Potentiometer is also connected to one of the leg bars to measure the FBEM angle.

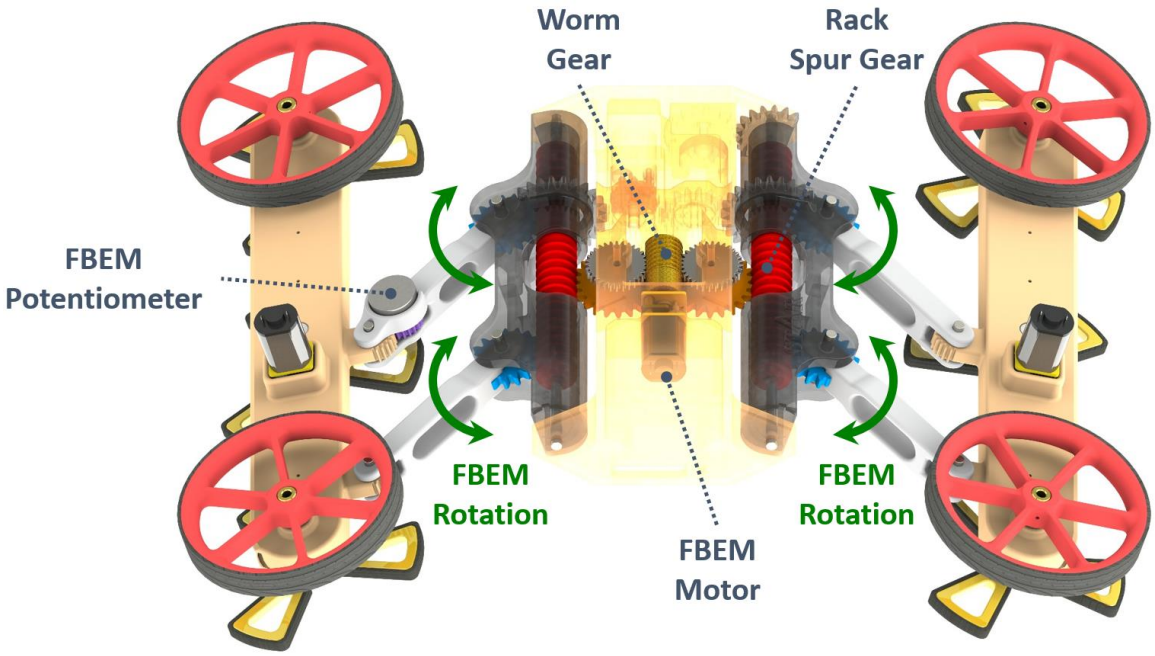


Figure 3.5 - The mechanical design of the FBEM. The motor rotation is converted to linear movement of the rack spur gear along the shaft which rotates the leg bars.

3.1.3 The Driving Mechanism

The RSTAR has a differential driving mechanism actuated by single motor on each leg as presented in Figure 3.6. Motor rotation is transmitted to the wheels using nine identical spur gears. An encoder is attached to each motor to measure the rotation rate of the wheels.

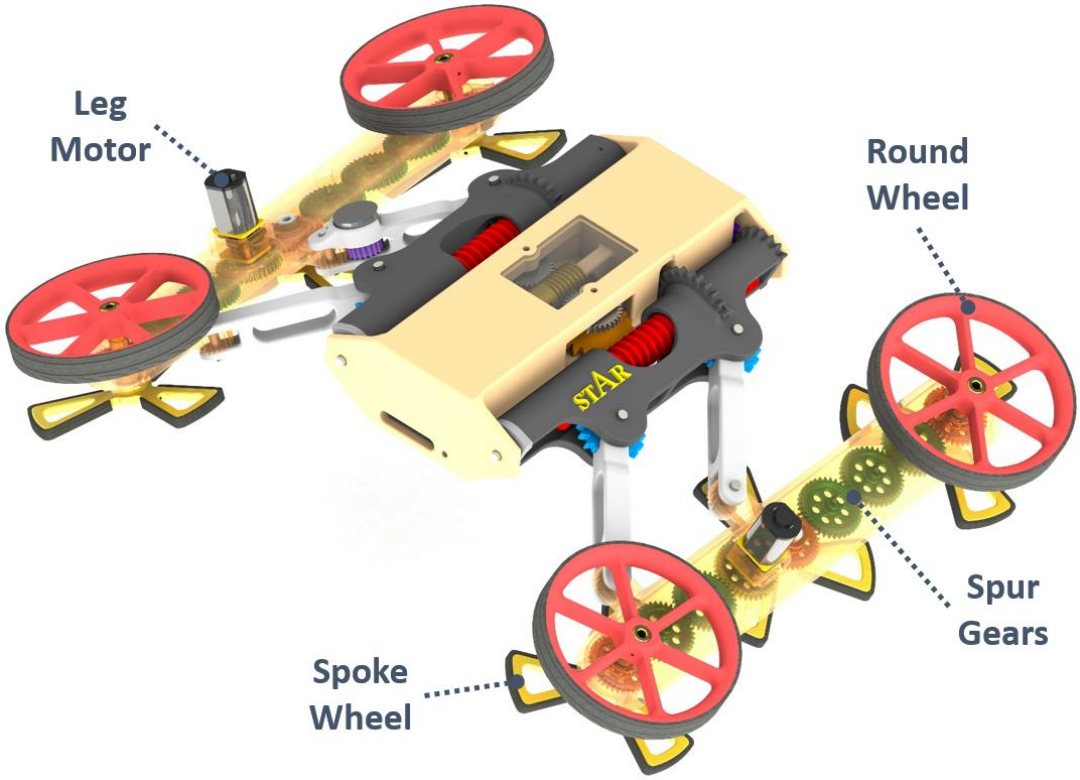


Figure 3.6 – The driving mechanism consists from DC motor, spur gears and wheels.

To improve stability and lower energy consumption, RSTAR can be fitted with wheels, spoked legs, or a combination of the two, giving it superior ability to engage different terrains. The round wheels fit for smooth surfaces while the spoke wheels are more efficient in unstructured environments. The RSTAR can flip its body upside down and drive inverted using dynamic maneuvers to change the type of wheels contacting the ground.

3.2 Robot Actuation

RSTAR is actuated using four motors. We used 12 mm diameter off-the-shelf motors (6-9 volts manufactured by Pololulu that are available with encoders which can be purchased at different gear ratios). The gearboxes with different gear ratios are of the same size, which simplifies their replacement without having to modify any other parts.

For higher driving speeds we used the lower gear ratio of 1:100 in the legs motors and for climbing we used a gear ratio of 1:300 (providing a torque of 0.18 Nm and 0.5 Nm respectively). The high ratio ensures high torque output and steady velocity. In the FBEM we used a 1:300 gear ratio, and in the sprawl mechanism a 1:1000 gear ratio was used for the high sprawling torque of 1.18 Nm. The robot is powered with three 3.7 Volts LiPo batteries; motors are powered with two 1000mAh batteries connected in series and the microcontroller is powered by a 400mAh battery.

3.3 Control System

The RSTAR can be controlled by either a human operator or by using a micro controller. In the human operator mode, a 2.4 GHz receiver is used to control the robot mechanisms. In the program mode, an off-the-shelf programmable Teensy 3.5 controller is used (32 bit, 120 MHz and compatible with Arduino libraries) and the robot is given a set of actions to perform. The main components of the control system are illustrated in Figure 3.7.

The sprawl and FBEM angles are controlled in a closed loop PD using the angular potentiometer attached to each mechanism. The rotational speed of the wheels is measured using magnetic encoders directly fitted to the shaft of the motors that provide 12 counts per motor revolution (1200 or 3600 counts per wheel revolution depending on the gear ratio of the motor).

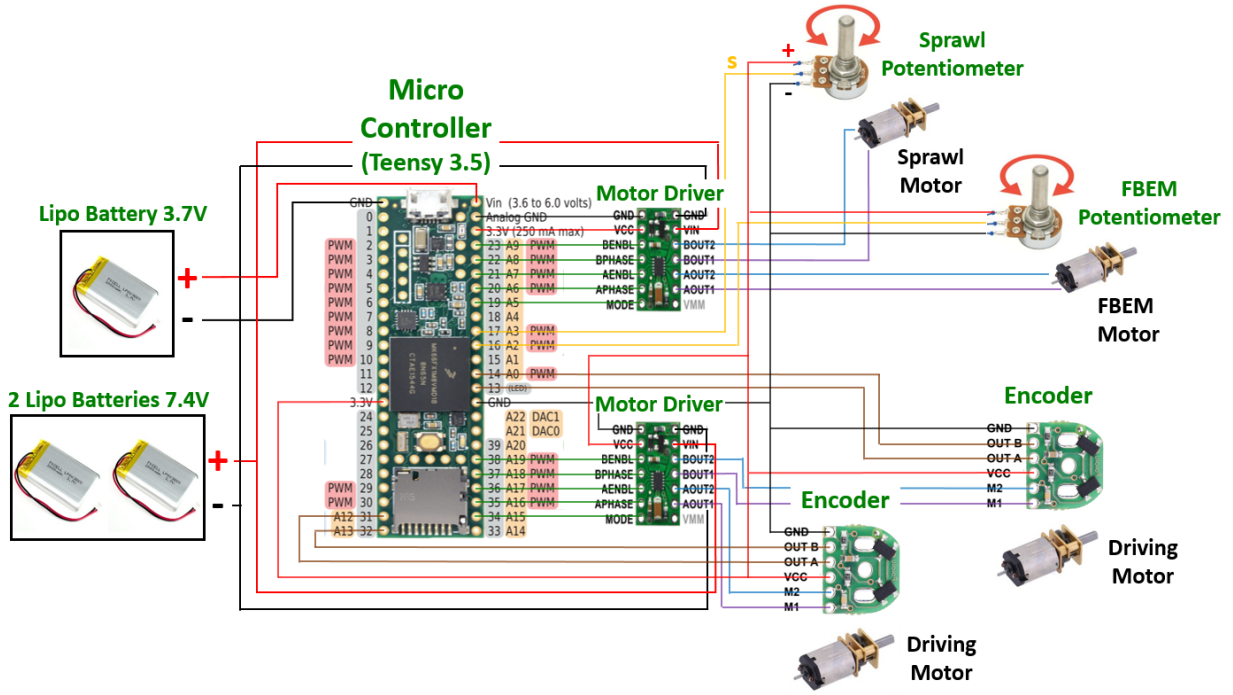


Figure 3.7 - The main components of the control system. The micro controller (Teensy 3.5) activates the motors using the motor drivers (H-bridge) and controls the RSTAR mechanisms in closed loop using the attached sensors (encoders and potentiometers). One battery supplies the voltage to the micro controller and another pair of batteries power the motors with increased voltage.

3.4 **Manufacturing**

The RSTAR mechanical parts are almost all manufactured with 3D printing. The components were printed by several 3D printing technologies depending on the required strength and the accuracy. Small parts or parts that required higher accuracy (like the gears or the robot body) were printed in PolyJet printing ("Objet Connex 350" 3D printer [A], "VeroWhitePlus" printing material [B]) or SLA printing ("FORMLABS FORM 2" 3D printer [C], Form resins [D]). Other parts were produced using FDM printing ("UPBOX" [E] or "Ultimaker 2" [F] 3D printers using PLA printing material [G]).

Considerable effort was made to simplify part replacement such as the motors, the bars of the FBEM, the spoke legs and wheels. Easy part replacement is essential for experimentation in different conditions and in case components are damaged during risky maneuvers.

Three versions of the RSTAR were manufactured as part of this dissertation and are illustrated in Figure 3.8. The main design differences are between version I and II, most of which concerned the sprawl mechanism and the FBEM. Version III is based on its previous version with adjustments made for adding angular potentiometers to the configuration mechanisms.

Figure 3.8 – The three versions of the RSTAR produced through this research.

4 Kinematic and Dynamic Analysis

In this section, we analyze the kinematics and dynamics of the robot. We present the different configurations that the robot can achieve and evaluate the torque requirements of the motors as a function of the external forces. This force and torque analysis was implemented during the design of the robot and motor choice (with a safety factor of 3-4).

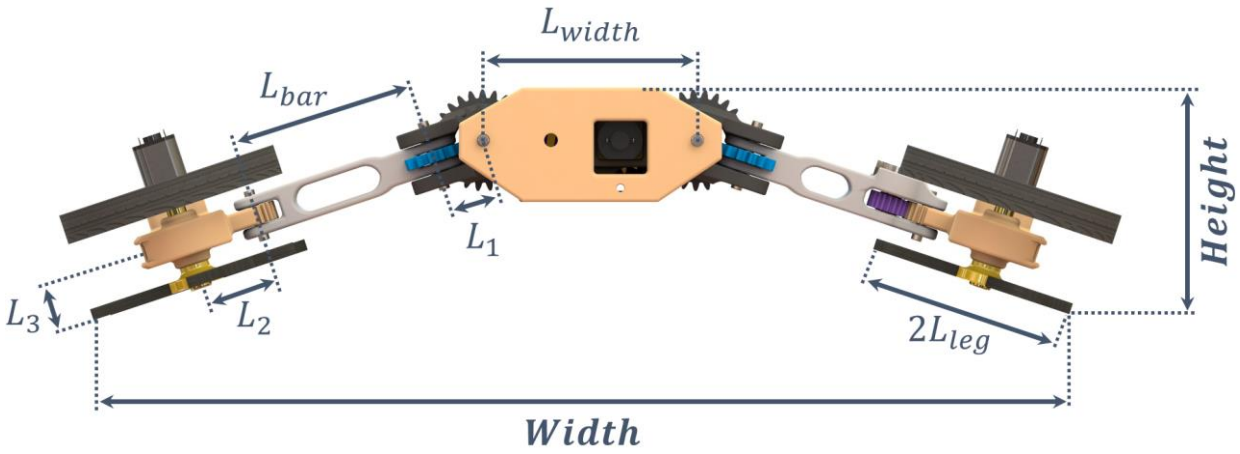


Figure 4.1 - Parameters for defining the RSTAR geometry.

4.1 Kinematic Analysis

The position of the legs of the RSTAR is a function of the sprawl angle and FBEM orientation. The work volume of the legs constitutes a two- dimensional shell as illustrated in Figure 4.2.

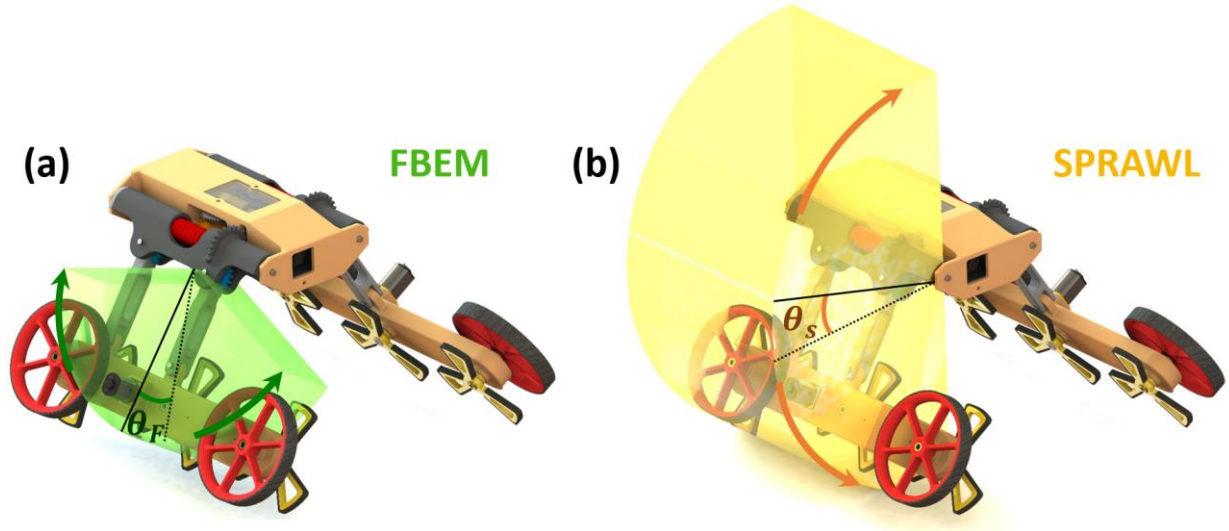


Figure 4.2 - The work volume of the RSTAR's legs consists of two shells: a) the work volume of the FBEM angle, b) the work volume of the sprawl.

The width of the legs (as defined in Figure 4.1) is:

$$width = L_w - L_3 \sin(\theta_s) + 2\left[L_1 + L_2 + L_{leg} + L_{bar}\cos(\theta_F)\right]\cos(\theta_s). \quad (5)$$

Because the robot can move the FBEM mechanism orientation from negative 72° to positive 72°, the legs can move in the fore-aft direction relative to the body by:

$$\Delta foreaft = 2L_{bar}\cos(\theta_F = 72°) = 1.9L_{bar}. \quad (6)$$

The height of the robot is:

$$height = \left(L_1 + L_2 + L_{leg} + L_{bar}\cos(\theta_F)\right)\sin(\theta_s) + L_h + L_3\cos(\theta_s). \quad (7)$$

Because the sprawl angle can be moved in the range of negative 90° to positive 90°, the tips of the legs can be moved in the vertical direction by:

$$\Delta height = 2\left(L_1 + L_2 + L_{leg} + L_{bar}\cos(\theta_F)\right)\sin(\theta_s = 90°). \quad (8)$$

4.2 The Mobility of the Center of Mass

The mobility of the center of mass (COM) can be used to enhance the stability of the robot and increase its maneuverability and ability to climb over obstacles. Raising and lowering the COM and moving it forward and backward relative to the legs can be used to flip the robot upside down and climb over a variety of obstacles.

In the fore-aft direction, the position of the COM is varied by activating the FBEM alone (see Figure 4.3). The core mass of the robot is on the main body, which we denote by m_{body} whereas the mass of each set of legs is m_{leg}.

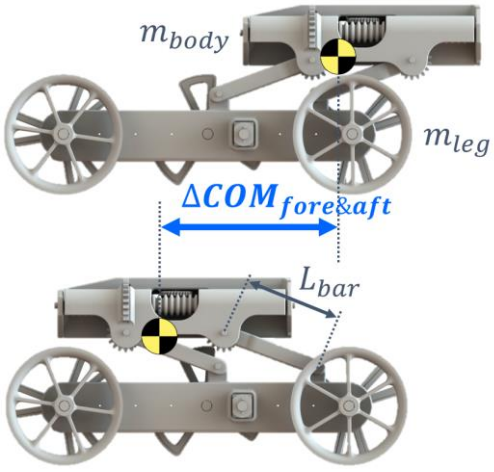


Figure 4.3 – The mobility of the COM in the fore-aft direction using the FBEM.

By denoting $\theta_{F-\max}$ as the maximum orientation angle of the FBEM and after neglecting the weight of the bars (nearly 3 grams each), the position of the center of mass is shifted forward and backward by:

$$\Delta COM_{foreaft} = \frac{2L_{bar}\sin\left(\theta_{F-\max}\right)m_{body}}{m_{body}+2m_{leg}}. \tag{9}$$

In the vertical direction, the COM can be moved from the in-plane configuration at zero sprawl (minimum height) to the 90 degrees sprawl configuration (maximum height) as illustrated in Figure 4.4:

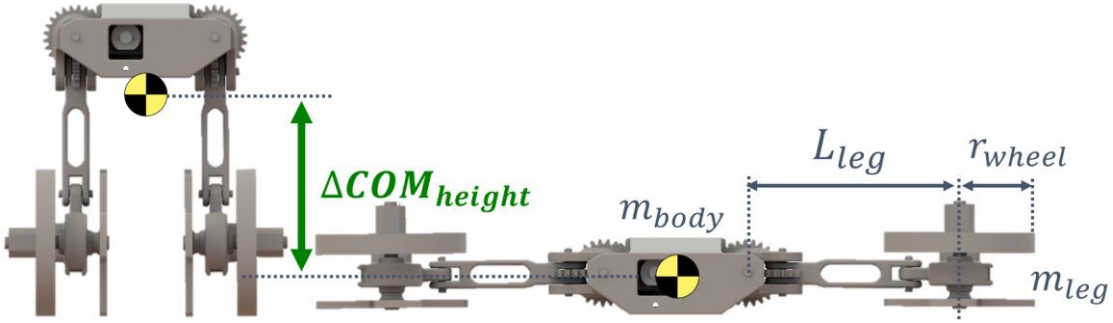


Figure 4.4 –The mobility of the COM in the vertical direction, can be moved by changing the FBEM and\or the sprawl mechanism.

The mobility of the COM in the vertical direction:

$$\Delta COM_{height} = \frac{\left(L_1+L_2+L_{leg}+L_{bar}\right)m_{body}}{m_{body}+2m_{leg}}. \tag{10}$$

For the given values of the actual robot: $m_{body} = 213$ grams, $m_{leg} = 104$ grams, $L_{bar} = 50$ mm, $L_{leg} = 29$ mm, $L_1 = 14$ mm and $L_2 = 20$ mm, the COM can be shifted in the fore-aft direction by 48.1 mm and in the vertical direction by 57.2 mm.

The angle Φ is the maximum tilt angle that the robot can statically withstand in the pitch direction before tipping over. Alternatively, by accelerating forward, the robot can pitch upward and flip itself upside down (see Figure 5.2). In addition, a climbing technique is based on pitching upward by accelerating, see Figure 5.7. The required acceleration *a* to pitch upward is:

$$a > g\tan\left(\Phi\right). \tag{11}$$

4.3 Force and Torque Analysis

In this section, we calculate the forces acting on the robot and the torques that must be provided by the different motors of the robot when moving on a horizontal surface and during climbing vertically between two walls. We analyze the cases where the robot works against gravity (raising its COM) which require larger torque forces. The absolute values of the forces acting on one side of the legs, in the normal, side, and fore-aft directions, relative to the body of the robot, are denoted by F_{normal}, F_{side}, and $F_{foreaft}$.

4.3.1 Moving Over a Horizontal Surface

A force diagram of the robot when moving on a horizontal surface is presented in Figure 4.5:

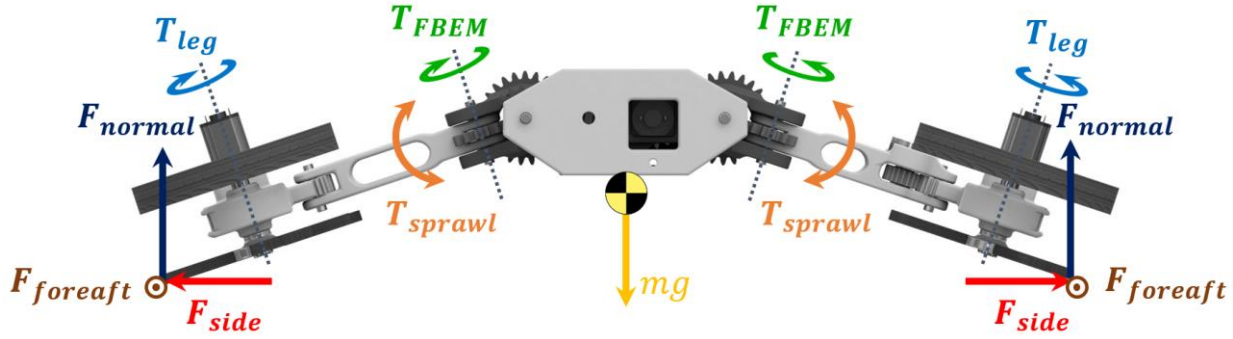


Figure 4.5 – The forces acting on the robot when moving over a horizontal surface. When lifting its body through the sprawl or the FBEM mechanisms, both the normal force and the friction side forces resist the motion.

When the robot lifts its body over a horizontal surface, either by increasing the sprawl angle or by extending its legs using the FBEM, and assuming low accelerations, the normal force F_{normal} is:

$$\left|F_{normal}\right| = \frac{mg}{2}. \tag{12}$$

Where m is the total mass of the robot. The side force F_{side} is:

$$\left|F_{side}\right| = \mu\frac{mg}{2}. \tag{13}$$

Note that F_{side} is pointed outwards when the robot increases the sprawl angle and inwards when it extends the length of its legs using the FBEM.

The torque acting on the sprawl joint T_{sprawl} is a function of the sprawl angle θ_s and the FBEM orientation θ_F .

$$\begin{aligned}\left|T_{sprawl}\right| &= \frac{mg}{2}\left[L_1 + L_2 + L_{leg} + L_{bar}\cos\left(\theta_F\right)\right]\cos\left(\theta_s\right) + \\ &+\mu\frac{mg}{2}\left[\left(L_1 + L_2 + L_{leg} + L_{bar}\cos\left(\theta_F\right)\right)\sin\left(\theta_s\right) + L_3\cos\left(\theta_s\right)\right]\end{aligned}, \tag{14}$$

Denoting L_t by the term:

$$L_t = L_1 + L_2 + L_{leg} + L_{bar}\cos\left(\theta_F\right), \tag{15}$$

And rearranging (14), we obtain:

$$\left|T_{sprawl}\right| = \frac{mg}{2}\left[\left(L_t + \mu L_3\right)\cos\left(\theta_s\right) + \mu L_t \sin\left(\theta_s\right)\right]. \tag{16}$$

Figure 4.6 presents the magnitude of sprawl torque required for lifting the body when the COF μ value is 0.3 (plastic contact with tile floor). The maximum value of T_{sprawl} is 258.5 Nmm and it obtained at $\theta_s = 16°$ and $\theta_F = 0$.

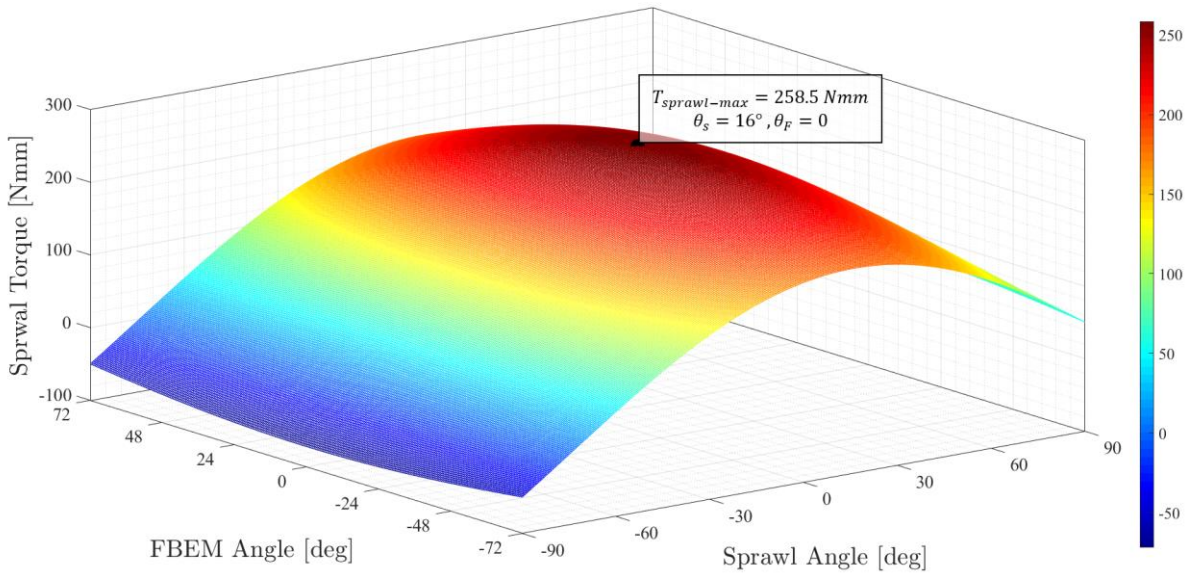


Figure 4.6 – The required sprawling torque for lifting the body in different sprawl and BEM angles.

The required torque by the FBEM when extending its legs is:

$$\left|T_{FBME}\right| = \frac{mg}{2}\left(\sin\left(\theta_s\right) + \mu\cos\left(\theta_s\right)\right)L_{bar}\sin\left(\theta_F\right). \tag{17}$$

Figure 4.7 presents the FBEM torque required when rising the COM assuming that the COF μ is 0.3 (plastic contact with tile floor). The maximum value of T_{FBEM} is 105.4 Nmm and it obtained at $\theta_s = 73.5°$ and $\theta_F = 72°$.

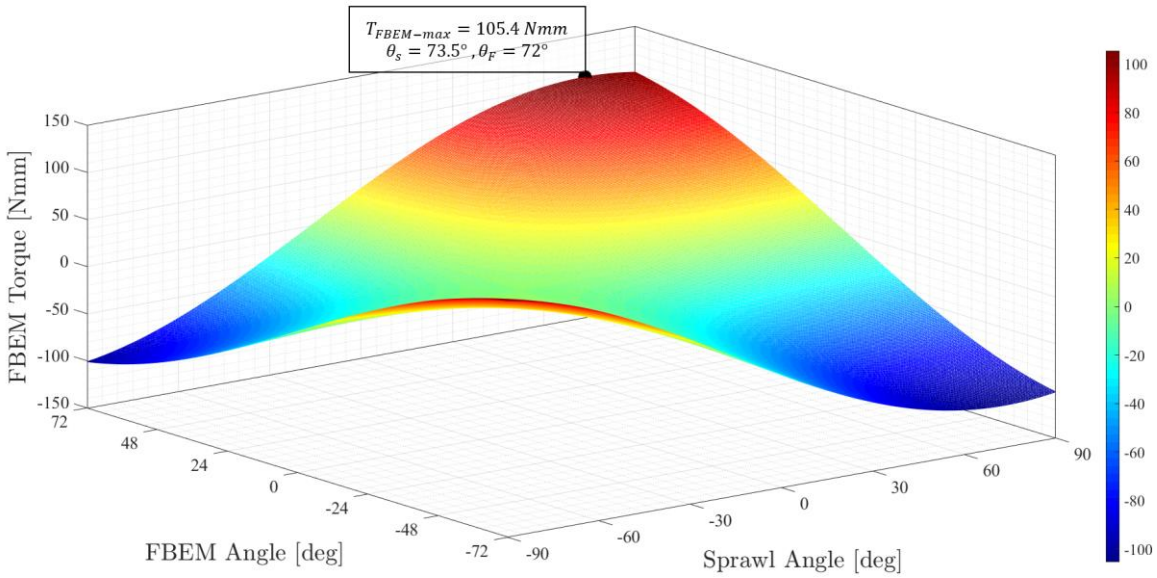


Figure 4.7 - The required FBEM torque when working against gravity in different sprawl and BEM angles.

4.3.2 Climbing Vertically Between Two Walls

Figure 4.8 presents a force diagram acting on the RSTAR when climbing vertically between two parallel walls:

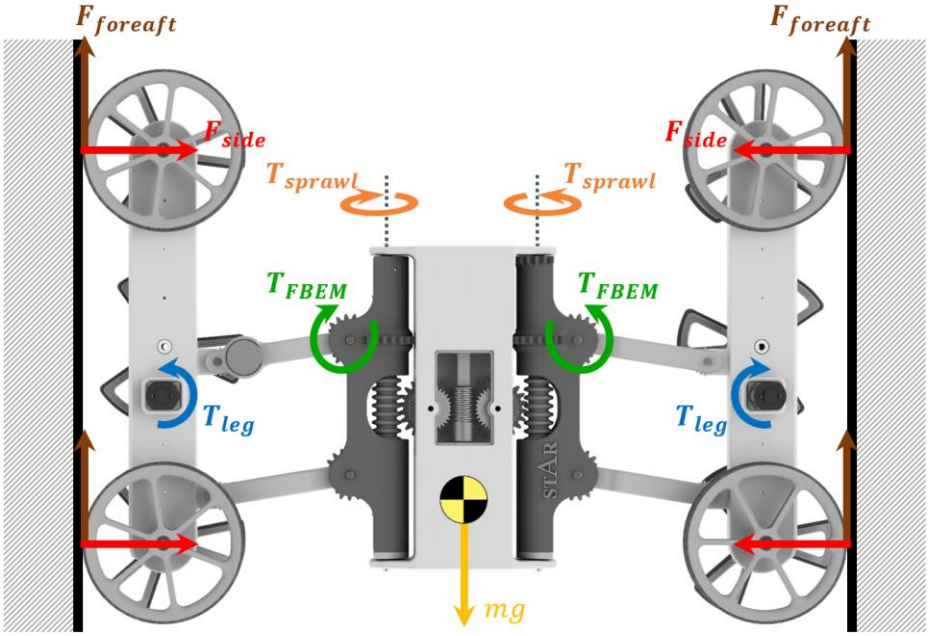


Figure 4.8 - The forces acting on the robot when while climbing vertically between two walls.

When the robot climbs vertically at constant speed inside a canal, the forces acting on the robot are in the fore-aft and side directions alone. The fore-aft force $F_{foreaft}$ that results from the friction force of the legs against the wall of the canal is equal to half of the weight:

$$\left|F_{foreaft}\right| = \frac{mg}{2}. \tag{18}$$

When climbing vertically between two walls, the robot must apply a side force:

$$\left|F_{side}\right| > \frac{mg}{2\mu}. \tag{19}$$

The torque that the sprawl mechanism must apply:

$$\left|T_{sprawl}\right| > \frac{mg}{2\mu}\left[\left(L_1 + L_2 + L_{leg} + L_{bar}\cos\left(\theta_F\right)\right)\sin\left(\theta_s\right) + L_3\cos\left(\theta_s\right)\right]. \tag{20}$$

The torque required by the FBEM is:

$$\left|T_{FBME}\right| = \frac{mg}{2\mu}\left(\sin\left(\theta_s\right) + \mu\cos\left(\theta_s\right)\right)L_{bar}\sin\left(\theta_F\right). \tag{21}$$

The torque acting on each set of legs, T_{leg}, during climbing in between two walls is:

$$\left|T_{leg}\right| = \frac{mg}{2}L_{leg}. \tag{22}$$

where L_{leg} is the radius of the wheels. Note that climbing is much easier with wheels rather than with spoke wheels.

5 Robot Locomotion Capabilities

This section presents the locomotion capabilities of the RSTAR. Here, the RSTAR was operated by remote control and was tested in variety of scenarios: running over different surface conditions, executing various maneuvers which included crawling over obstacles, climbing between two walls, and demonstrating the turtle locomotion gait in which the robot can move without rotating its legs (see video [24]).

5.1 Running over a Variety of Surfaces

We tested the robot outdoors on a variety of surfaces, see Figure 5.1. The robot successfully crawled over gravel and even climbed a small rocky incline. The robot also crawled successfully over grass and rough sandy surfaces and climbed over concrete.

Figure 5.1 – RSTAR crawling on variety of surfaces including gravel, soft ground, leaves and grass (see video [24]).

5.2 Inverted Running, Combining Wheels and Spoke Wheels

RSTAR can flip itself upside down and vice versa without external intervention as illustrated in Figure 5.2. This feature can be used to decrease its cost of transport and reduce oscillations by fitting its legs with regular wheels on one side for running over smooth surfaces and fitting spoke legs on the other side for running over unstructured terrains. Note that inverting

the body will bring the outside swing of the legs in contact with the ground, resulting in the original direction drive. In this double inversion, all control laws are consistent and the leg drive and steering control continue to function as expected.

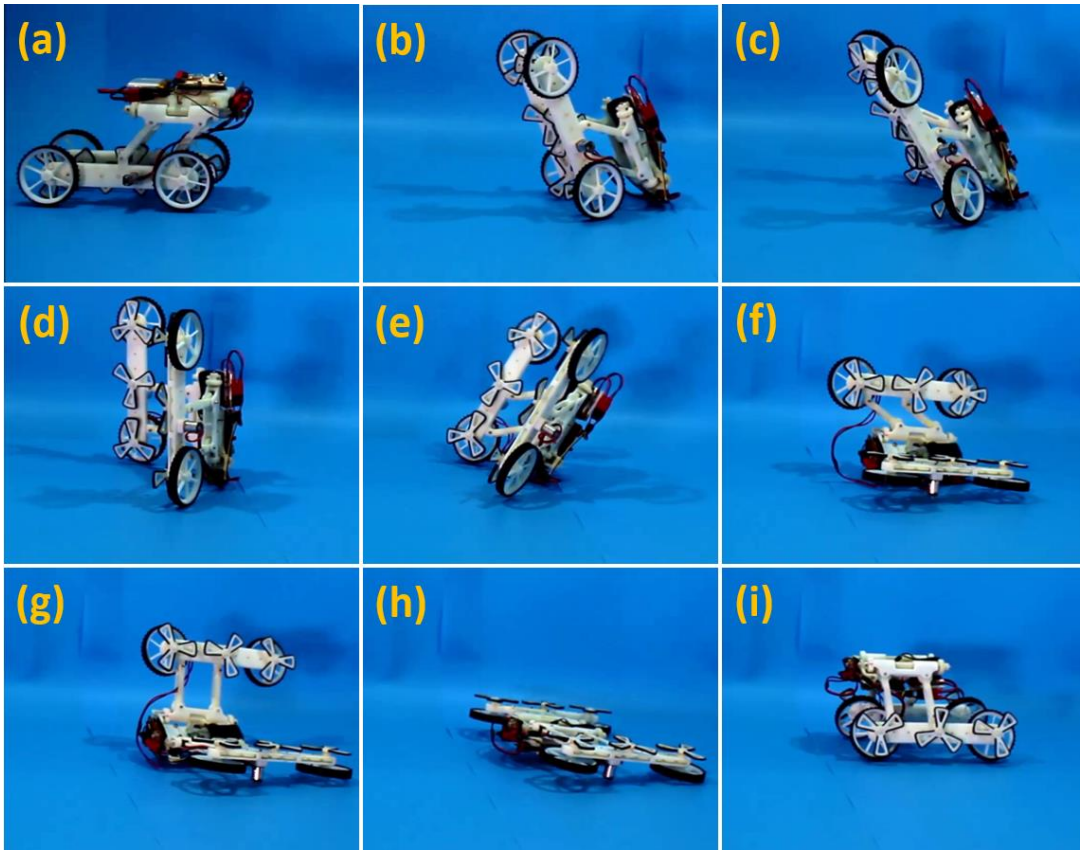


Figure 5.2 – RSTAR can flip itself upside down so that it can be driven on one side with wheels over flat surfaces and the other side with spoke wheels over challenging surfaces in unstructured environments (see video [24]).

Self-flipping is executed by changing the position of the COM, as shown in Figure 5.2. First, the RSTAR accelerates when its COM is backward (5.2.a) and by immediate braking, its body falls backward (5.2.b). By flattening its sprawl angle (5.2.c) and moving its COM backward using the FBEM (5.2.d-5.2.e) the robot flips backward (5.2.f). After the robot is inverted, actuating the sprawl mechanism lifts the body upwards and makes the wheels contact the ground (5.2.g-5.2.i). The result of this process is that the wheels change from spoke legs (5.2.a) to regular wheels (5.2.i).

In previous work [8] we showed that the spoke wheels at a low sprawl angle (15 degrees) have a mechanical COT (not including electrical losses) of nearly 0.2. Although this COT is relatively low compared to robots at this scale, it remains two orders of magnitude higher than the rolling friction (or rolling resistance) of the wheels. Therefore, to reduce the COT (and

therefore extend the working range of the robot) and to reduce the vibrations resulting from the spoke wheel collisions with the surface, the robot can be run when inverted over smooth surfaces such that the regular wheels engage the surface.

5.3 Turtle Locomotion Gait

One of the unique locomotion gaits that the RSTAR can perform is a turtle-like locomotion in which the robot advances without rotating its wheels [25] (somewhat similar to inchworming [26]). While the turtle gait is a slow method of crawling, it is very effective on soft and slippery ground and when crossing canals.

This gait is made up of a sequence of four steps and is done by activating the sprawl angle and the FBEM without driving the wheels as demonstrated in Figure 5.3. Starting in an almost flat configuration (5.3.a), the body is lifted until it no longer touches the ground using the sprawl mechanism (5.3.b). Then the body is pushed forward (5.3.b-5.3.d) using the FBME. In (5.3.f), the robot lifts its legs using the sprawl mechanism. Once the legs are in the air, the robot moves its legs forward using the FBME (5.3.f-5.3.h). Finally, the robot pushes its legs downwards (5.3.i) to complete a full turtle gait cycle. In our experiment consisting of 6 cycles, the robot advanced 60 cm in 1:50 minutes at a rate of 10 cm/cycle.

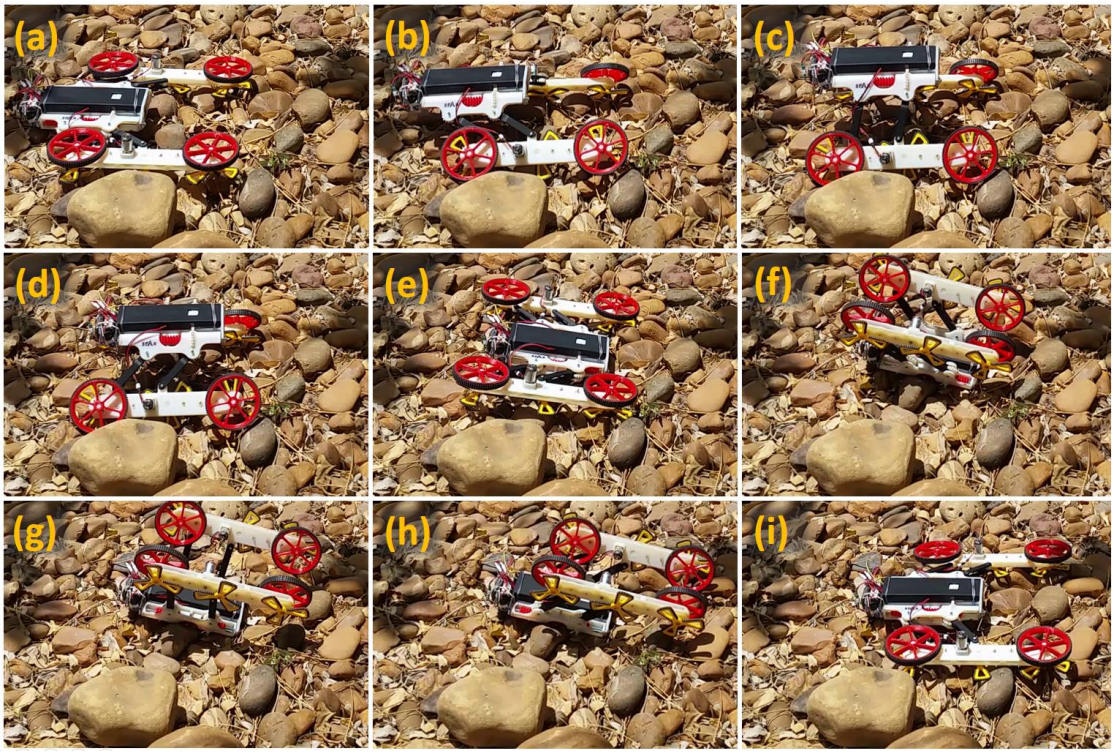


Figure 5.3 – Full cycle of turtle gait locomotion, the RSTAR is advancing forward using it legs without rotating its wheels (see video [24]).

5.4 **Vertical Climbing**

RSTAR was designed for easy motor replacement. In our horizontal experiments we used a 100:1 gear ratio which allows the robot to run at a maximum speed of nearly 1m/s. The nominal thrust force at a 1:100 gear ratio is 6.4 N, which theoretically speaking, is sufficient for climbing vertically (almost twice the weight). However, due to internal friction losses which increase substantially during climbing because of the normal forces that must be applied to the walls, we had to increase the gear ratio to 1:300. At this ratio, the horizontal speed drops to 35 cm/s but the thrust force increased to 17 N and the robot successfully climbed when placed vertically at 20 cm/s (see Figure 5.4). Note that although the robot can pitch upwards and change its width to touch the two sides of the wall, we had to place the robot vertically between them. We believe that the transition from horizontal locomotion to vertical climbing is feasible and will be the focus of our future research.

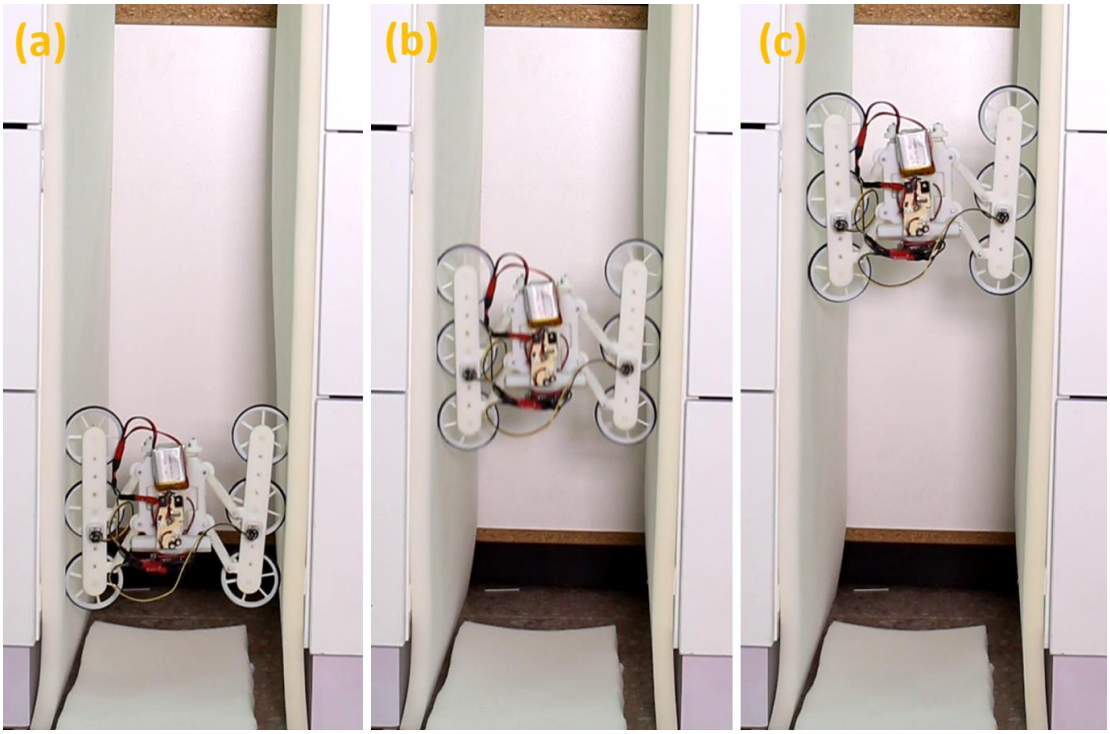


Figure 5.4 - When placed between two walls, the RSTAR fitted with wheels can climb at 20 cm/s. The RSTAR's width can be varied to touch both sides of the walls (see video [24]).

5.5 Crawling Between Two Walls

RSTAR can also crawl horizontally between two parallel walls (see Figure 5.5) by applying enough pressure on the walls with its wheels using the sprawl mechanism and the FBEM. It can also switch from horizontal crawling to vertical climbing and vice versa. This feature is very useful for movement inside pipes and can also be used for skipping over obstacles as illustrated in Figure 5.5.

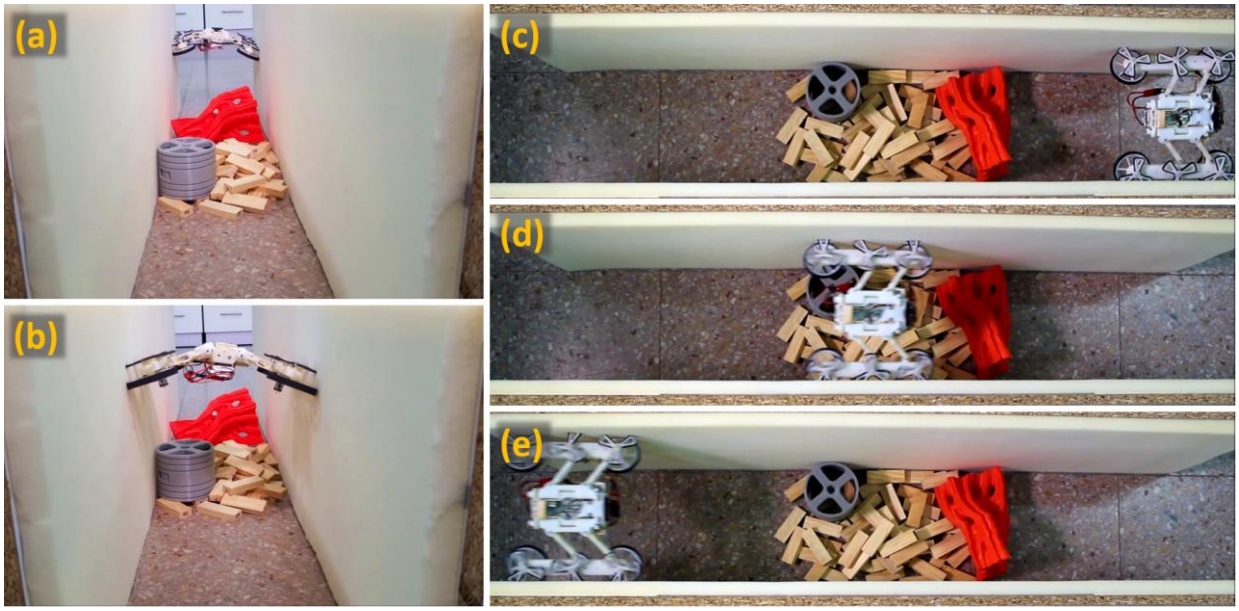


Figure 5.5 – horizontal crawling between two walls, using the sprawl and FBEM RSTAR can adjust its width and create enough pressure with the walls (see video [24]).

The transformation from horizontal crawling to vertical climbing can be done by moving the COM position using the configuration mechanisms (sprawl and FBEM) which change the pitch angle of the robot while it continues to advance. Note that a necessary condition to prevent the RSTAR from losing contact with the walls and fall down' it has to keep its width constant while changing the COM position. Currently the lab is working on developing control system that can control the configuration mechanisms in the case of movement in curved pipes.

5.6 Overcoming Obstacles

The ability of the RSTAR to reconfigure its shape and move its COM impart the RSTAR with improved ability to overcome different types of obstacles. It can crawl through low passages by minimizing its height to 49 mm by changing the sprawl angle to a nearly flat configuration (a minimum of 3 to 5 degrees sprawl is required to ensure it can advance). By narrowing its width up to 115 mm, the RSTAR can also traverse narrow passages. In addition, the RSTAR is very efficient in climbing over obstacles of up to 6.5 cm by implementing different techniques using the sprawl and FBEM mechanisms combining shape configuration and COM mobility.

5.6.1 Turtle Gait Climbing

The turtle gait an efficient way to climb over obstacles, In Figure 5.6 the turtle gait is demonstrated, climbs over a 53 mm obstacle with 58 mm diameter wheels.

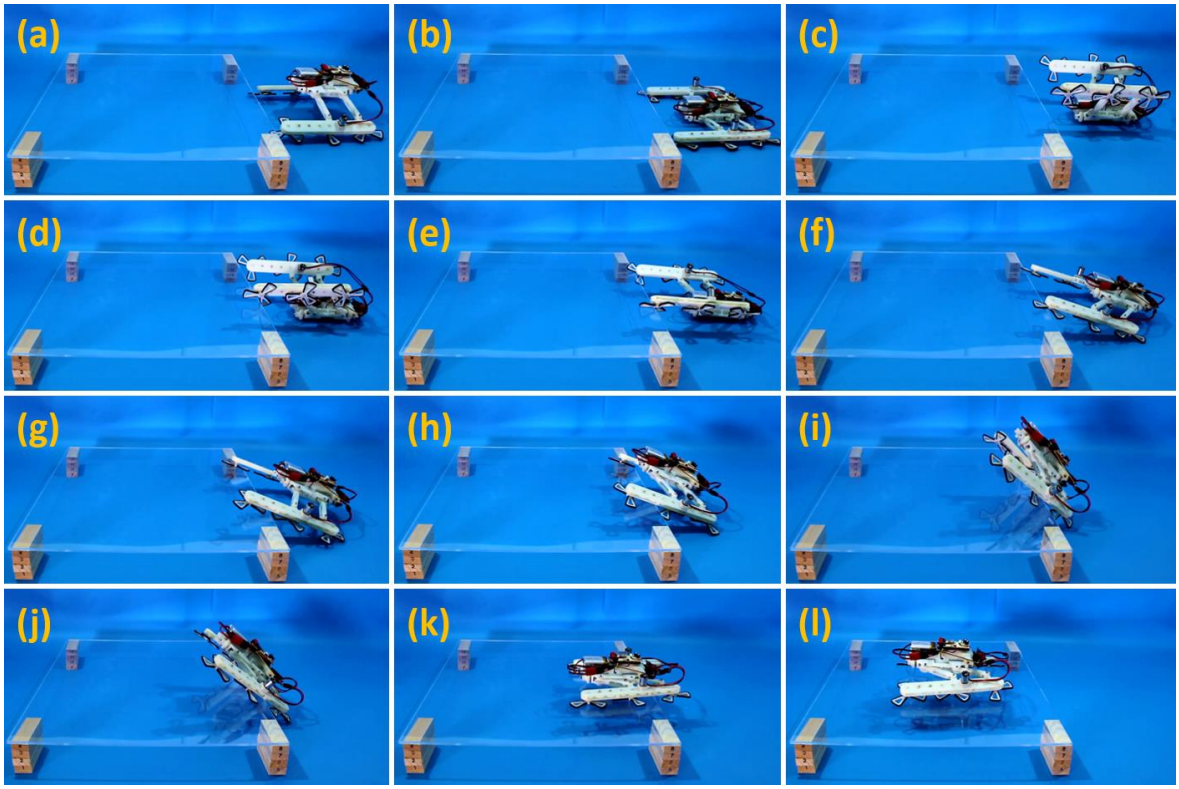


Figure 5.6 - RSTAR climbing over an obstacle using the turtle locomotion gait which is achieved by the actuation of both the sprawl angle and the four bar extension mechanisms (see video [24]).

After the robot is positioned in front of the step (5.6.a), its legs are lifted using the sprawl mechanism (5.6.a-5.6.c) until its front wheels are located above the step. While the legs are in the air, the legs are oriented forward as far as possible using the FBME (5.6.d-5.6.e). Then, using

the sprawl mechanism the body is lifted and the front wheels touch the step (5.6.f) until the RSTAR leans on the step in a stable fashion. In (5.6.f-5.6.h) the body is moved forward using the FBME. The movement of the COM forward enables climbing by rotating the wheels forward (5.6.i) and by lowering the COM with the sprawl mechanism (5.6.j) the robot falls forward (5.6.k) and is able to advance on the step using its wheels (5.6.l).

5.6.2 Pitching Upward for Climbing

The robot can reach the tip of the obstacle by pitching its body upward and advancing towards the obstacle see demonstration in Figure 5.7. Starting from (5.7.a), the robot raises its body (5.7.b) and then accelerates (5.7.c) to pitch its body upwards (5.7.d). Using the spoke wheels, the robot advances to the obstacle (5.7.e) and by reducing the sprawl it falls on the obstacle (5.7.f). At this point, the robot moves its COM forward (5.7.g) and drives its spoke wheels forward to complete its climb (5.7.h).

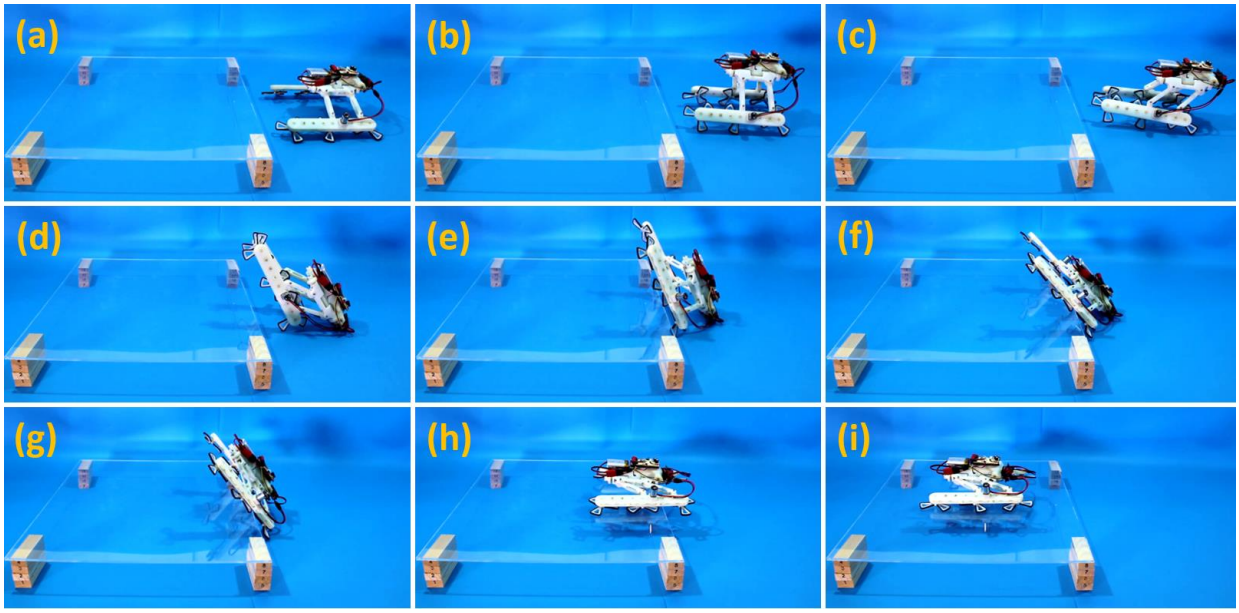


Figure 5.7- The robot is climbing on top of the obstacle by pitching its body upward and then moving its COM across the edge of the obstacle (see video [24]).

6 Implementation of Reinforcement Learning on the RSTAR

This section details the simulated virtual environment used for the learning process, the different obstacles that the robot learned to overcome, the discretization of the state space, and the learning algorithm.

6.1 Simulation Environment and Obstacle Definition

The simulations were conducted using a Unity® software environment (real-time engine development platform) [47] and included a (1:1 ratio) model of the RSTAR and the different obstacles. We defined the RSTAR's kinematics and tuned its speed and contact properties with the surface to mimic those of the physical robot.

Three common use cases in which the robot can overcome an obstacle were learned in three separate simulations: (1) A narrow (180 mm wide) channel in which the robot has to reduce its width to pass through (Figure 6.1– a). (2) A low entry (55 mm high) where the robot has to lower its body to crawl underneath (Figure 6.1- b). (3) A 50 mm step obstacle the robot needs to climb over by shifting its center of mass (Figure 6.1 – c). The geometric properties were modifiable so that the simulation results could be tested for relevancy on obstacles of different sizes. The simulations were performed on an Intel® Core™ i7-3632QM processor 2.2GHz, 8GB RAM memory running a Windows 10 operating system x64 bits.

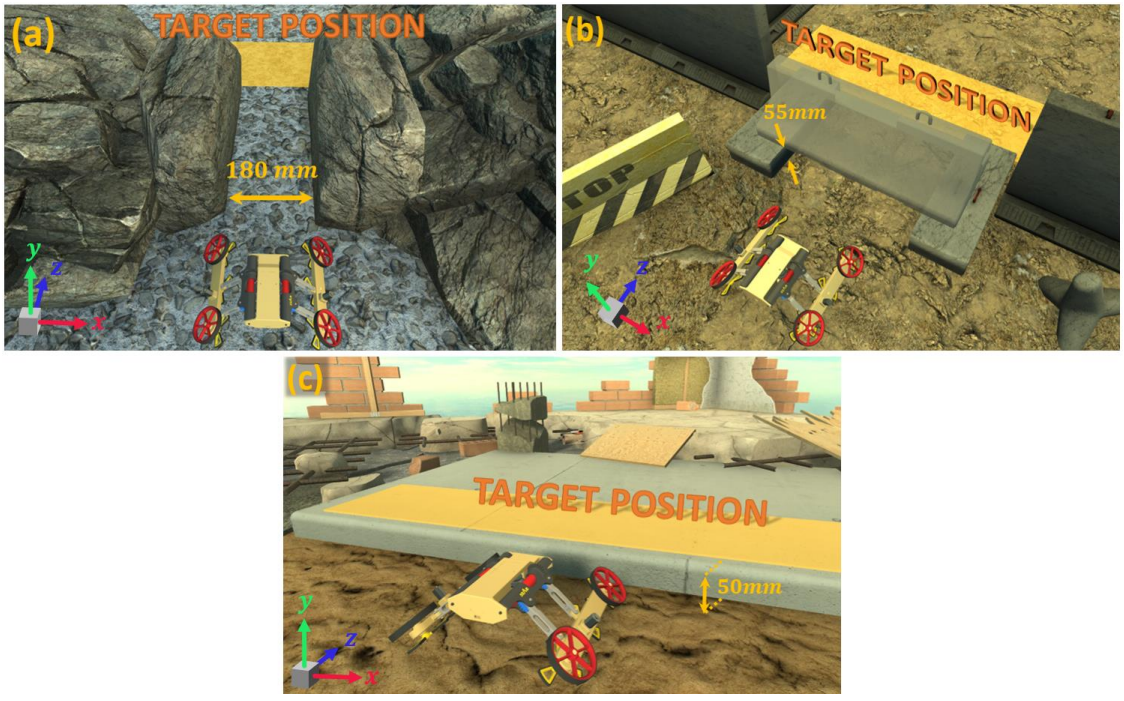


Figure 6.1 - The different obstacle use cases that RSTAR learned (a) A squeezing through a 180 mm channel. (b) Crawling underneath a 55 mm high opening. (c) Climbing over a 50 mm high step.

6.2 Learning Procedure

The *Q*-learning algorithm was applied using the RSTAR Unity® model. Figure 6.2 illustrates the iterative procedure of the learning process: the RSTAR observes its state, *s*, performs the action, *a*, and receives a reward, *r*. Its next state, *s'*, is produced by the environment, and finally the Q value of the related action-value pair is updated. Each learning iteration ends when one of the following conditions is met:

1) The RSTAR successfully reached its target.

2) The number of actions performed exceeded the maximal number of actions allotted.

3) The RSTAR reached a "dead end" by moving sideways past obstacle borders.

4) The RSTAR flipped over.

All the intermediate learning data including states, actions, rewards, and Q matrix values, were retained for convergence analysis and validation.

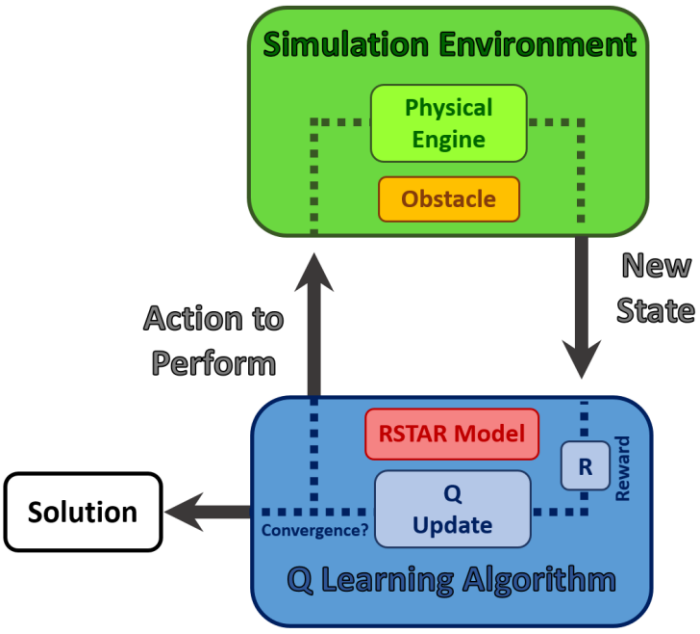


Figure 6.2 -The learning process using the physical Unity® engine and Q-learning algorithm.

6.3 States

The state of the agent (the RSTAR) was based on its position, orientation, and configuration. To reduce the dimensionality of the state space, the position of the robot was limited to the advancing direction (z) alone and roll was neglected. Altogether, the agent's state was composed of five parameters: a) the position in the advancing direction, b) the yaw, c) the pitch, d) the FBEM angle, and e) the sprawl angle, as illustrated in Figure 6.3.

STATES	ACTIONS
ADVANCING DIRECTION	**1** - MOVING FORWARD
YAW	**2** - MINUS FBEM
PITCH (Climbing on only)	**3** - PLUS FBEM
FBEM	**4** - MINUS SPRAWL
SPRAWL	**5** - PLUS SPRAWL

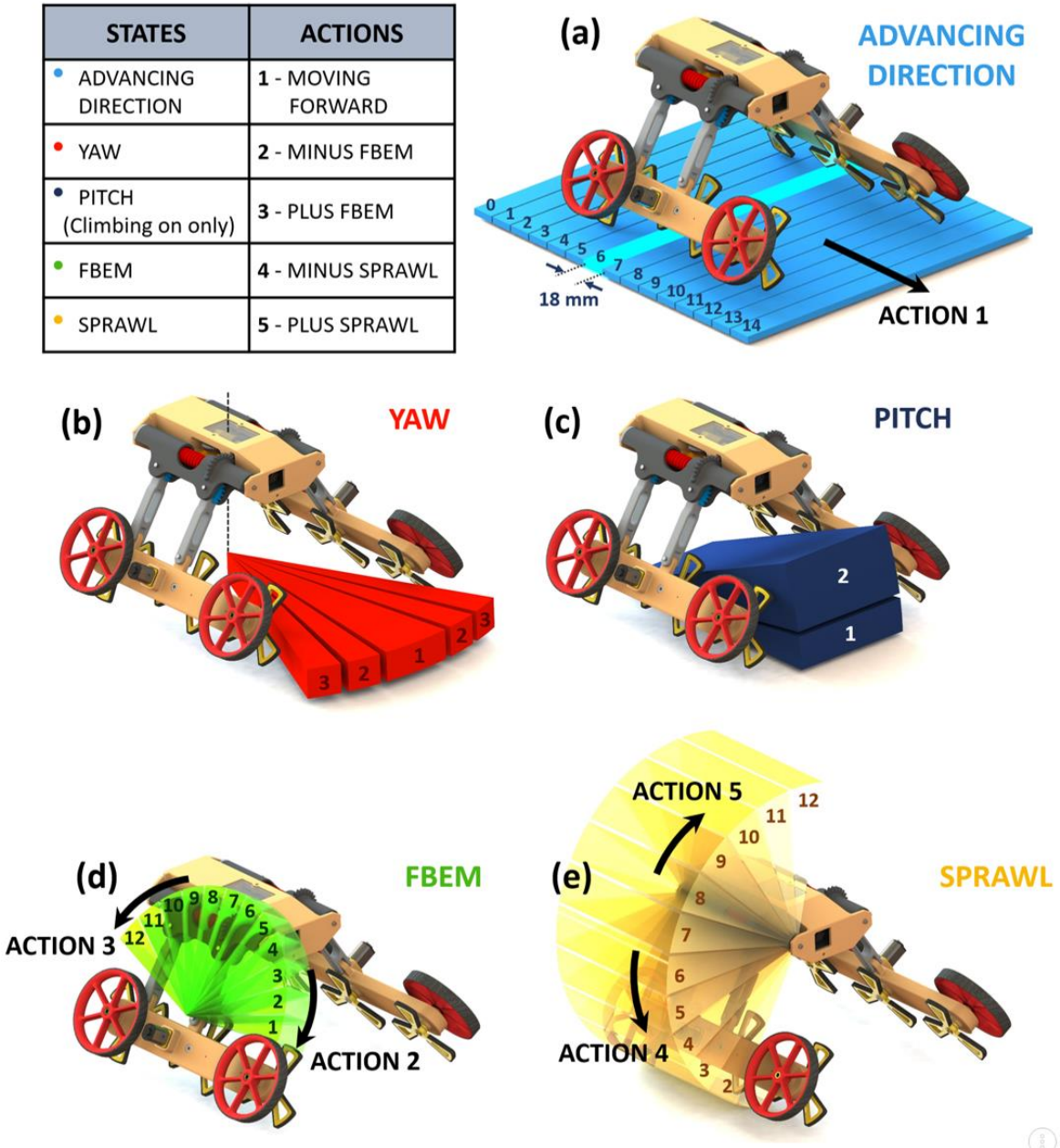


Figure 6.3 - The partition of the state and action spaces into discrete values. Partition of sprawl was adapted to the required sensitivity of the use case.

The partition of the state and action spaces into discrete values was slightly different as a function of case. The partition of the advancing direction and the FBEM was constant and equal to 18 mm and 12^o respectively in all three simulations. The yaw angle, α, was divided into three angle ranges in all simulations: $0 \le |\alpha| < 5$, $5 \le |\alpha| < 10$, and $10 \le |\alpha|$. The sprawl angle was partitioned into step sizes of 9^o or 15^o depending on the use case and required sensitivity (Table 1). Because the first two problems (squeezing through a channel or a low opening) did not involve climbing, the pitch was ignored. In the third problem in which the robot needed to climb over the step, the pitch angle, β, was divided into two ranges, $0 \le \beta < 5^o$, $5^o \le \beta < 180^o$.

6.4 Actions

The action space consisted of five actions (Figure 6.3): 1) Rotate the wheels forward by one third revolution (60.7 mm). 2-3) decrease the FBEM angle (2) or increase it (3). 4-5) increase (4) or reduce (5) the sprawl angle. Note that the robot might not be able to perform a specific action if prevented to do so by an obstacle. For example, in Action type 1, the wheels simply slide if the robot cannot advance forward.

Table 1 - Number of states and actions in each simulation

Use case / Partition number and size	Through a channel	Low opening	High obstacle
Position (z direction)	43	34	24
Size:	18 mm	18 mm	18 mm
Yaw Angle	3	3	3
Range:	$0 \le \|\alpha\| < 5$, $5 \le \|\alpha\| < 10$, and $10 \le \|\alpha\|$		
Pitch Angle	-	-	2
Range:	$0 \le \beta < 5^o$, $5^o \le \beta < 180^o$		
FBEM	12	12	12
Size:	12^o	12^o	12^o
Sprawl	12	20	12
Size:	15^o	9^o	15^o
Possible Actions	5	5	5
Q matrix cells	92,880	122,400	103,680

6.5 Q Matrix initiation

Each action-state pair was rated based on the expected value of taking the action when in that state. The Q matrix was 5D for the use cases of squeezing through a channel and ducking underneath an obstacle and 6D for the climbing over the obstacle use case. The number of Q matrix cells was on the order of 10^5 (Table 1).

To encourage the robot to advance towards the obstacle and overcome it, the initial Q matrix values were based on the advancing direction and the yaw angle. The values increased as a function of the advancing direction and decreased with the absolute value of the yaw. In all use cases, the workspace was divided into four zones (i-iv) along the advancing direction (z). Figure 6.4-a presents the specific zone borders of the climbing over an obstacle use case. The other use cases had a similar pattern but different border values.

Zone (i): Located behind the agent's initial position ($z \le 3$) and was characterized by negative equal values for all yaw angles:

$$Q_i(s_n, a_n) = -100\left(Z_{tot} - z\right). \tag{23}$$

where Z_{tot} is the total number of partitions in the z (advancing) direction and $z \le 3$. This zone could be reached if the RSTAR moved its COM backwards using the FBEM.

Zone (ii): Located between the agent's initial position and proximity line (4 bins before the success line $3 \le z < 15$). In this zone, the Q started with positive values and monotonously increased as the robot advanced towards the success zone. Lower values were given to greater yaw angles between 5° to 10° and even lower to values beyond 10° to encourage the robot to maintain a straight orientation.

$$Q_i(s_n, a_n) = \begin{cases} 100z & \left|\theta_{yaw}\right| \le 5 \\ 5z & 5 < \left|\theta_{yaw}\right| < 10 \\ z & 10 \le \left|\theta_{yaw}\right| \end{cases}. \tag{24}$$

Zone (iii): In proximity to the success zone, with length in 4 partitions (in the advancing direction $15 \le z < 19$) and Q values that were substantially higher than zone (ii). I this zone, The Q values were independent of the yaw:

$$Q_i(s_n, a_n) = 1000z. \tag{25}$$

Zone (iv): The success zone, characterized by the highest initial Q values (identical to Eq.25) to ensure that the robot was attracted to the success zone and remained in it.

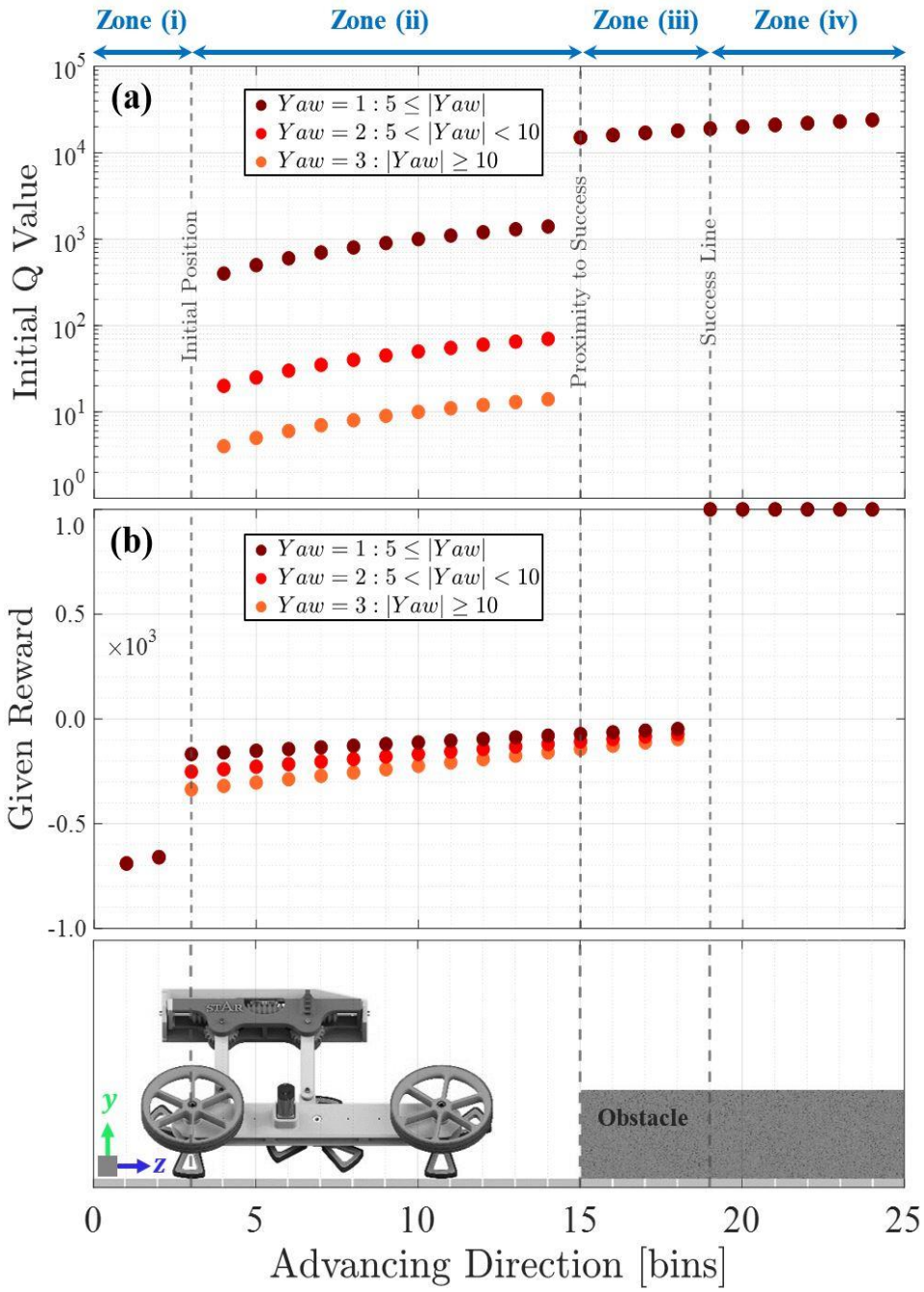


Figure 6.4 - Given reward (a) and initial Q value (b) as a function of the advancing direction and yaw angle.

6.6 Reward and Update

As in the Q matrix initialization, the workspace was divided into the same four zones (Figure 6.4-b). The rewards were negative in the first three zones to minimize the number of steps and ensure convergence of the solution to an optimal solution:

Zone (i): A highly negative rewards zone.

$$r(s_n, a_n) = -30\left(Z_{tot} - z\right). \tag{26}$$

Zone (ii): The reward in this zone was also negative but monotonously increasing to "encourage" the agent to leave the zone and advance forward. The reward was more negative if the yaw angle was larger than 5^o to ensure that the agent did not attempt to rotate sideways.

$$r(s_n, a_n) = \begin{cases} -8\left(Z_{tot} - z\right) & \left|\theta_{yaw}\right| \le 5 \\ -12\left(Z_{tot} - z\right) & 5 < \left|\theta_{yaw}\right| < 10 \\ -16\left(Z_{tot} - z\right) & 10 \le \left|\theta_{yaw}\right| \end{cases}. \tag{27}$$

Zone (iii): The reward function in this zone was similar to zone (ii), and was designed to ensure it continued attempting to advance forward.

Zone (iv): the success zone was highly positive to ensure that the agent advanced towards it and finished its climbing learning.

$$r(s_n, a_n) = 30000. \tag{28}$$

The Q matrix was updated using:

$$\mathrm{Q}(s_n, a_n) \leftarrow (1-\alpha)\mathrm{Q}(s_n, a_n) + \alpha[r_n + \gamma \max_{a'} \mathrm{Q}(s_{n+1}, a')]. \tag{29}$$

where r_n was the reward for taking action a_n at state s_n, s_{n+1} was the state of the agent after taking the action a_n. The action a' was the one associated with the highest possible $\mathrm{Q}(s_{n+1},a)$ value. The learning rate α was 0.6 whereas the discount factor γ was 0.9.

6.7 **Policy**

The learning was conducted as an ε-greedy policy where the probability of taking a random action, ϵ, decreased by:

$$\varepsilon \leftarrow \varepsilon - \frac{\varepsilon_i - \varepsilon_{\min}}{N_{\max}}. \tag{30}$$

Where ε_i is the initial probability, $\varepsilon_{\min}$ is its minimum value and $N_{\max}$ is the total number of times random actions is taken. In all simulations we used: $\varepsilon_i = 1$, $\varepsilon_{\min} = 0$ and $N_{\max} = 30000$.

7 Simulation Results

In this section, we first present the convergence rate of the algorithm. Then an analysis of the solution and its validity over a range of parameters is presented. Finally, a comparison between the simulation and human operators is presented.

7.1 The Convergence Rate

The convergence rate of each simulation was tested using three parameters: the sum of reward over a path received at every iteration, the number of actions per iteration, and the success rate of the iterations. The results of squeezing through a channel, ducking underneath an obstacle and climbing on top of an obstacle are presented on Figure 7.1-Figure 7.3 respectively.

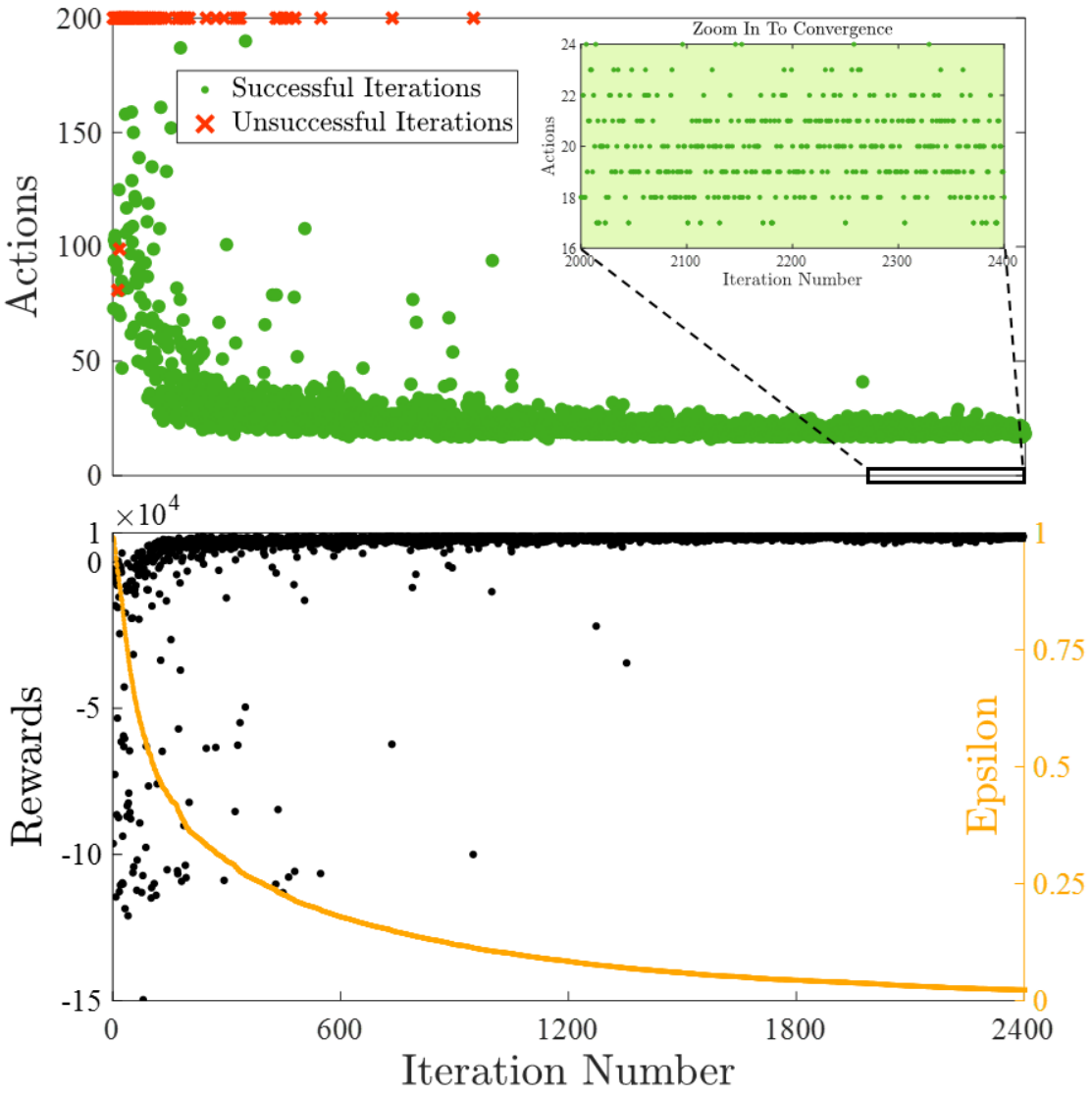


Figure 7.1 - The number of actions in each iteration (top); The sum of the rewards and value of ε in each iteration (bottom) of the squeezing through a channel use case.

The algorithm started to find feasible solutions after 100 iterations (60% success) (Figure 7.1, top). Following that, the number of actions of the found solution decreases and after 600 iterations, the success rate is nearly 100%. The quality of the solution (the number of actions) improve with the number of iterations and reaches an average of 20 actions per solution. With the best solution (the smallest number of actions) performed with 17 actions. In section IV.E, we will see that some of the solutions with a larger number of actions had advantages for overcoming challenging obstacles not previously learned. The iterations started with a highly negative sum of rewards (Figure 7.1, bottom) and as the iterations advanced, the sum of the rewards increased (the absolute decreased) as the robot succeeded in finding feasible solutions after nearly 200 iterations and then continued to slowly improve.

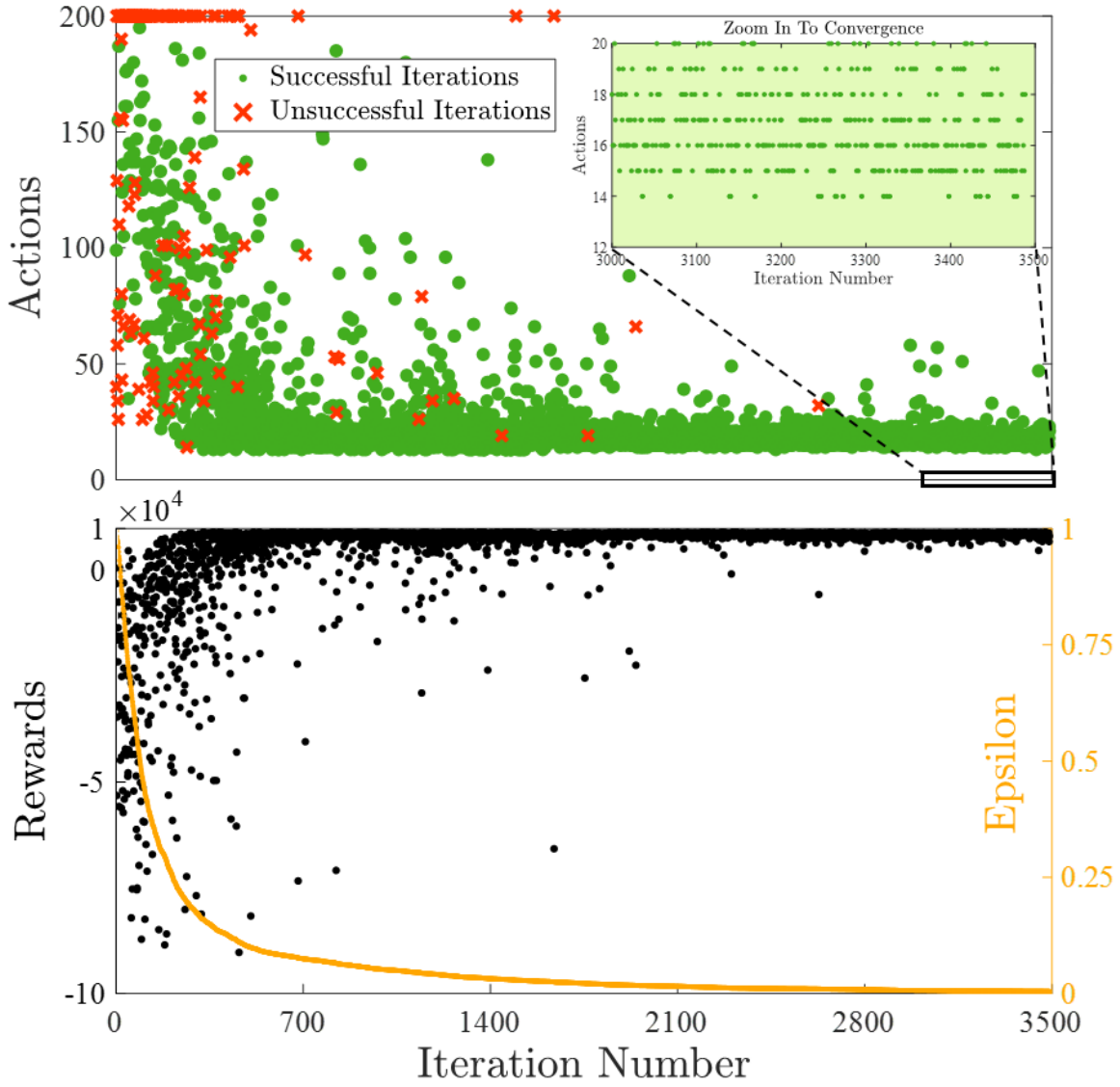


Figure 7.2 - The number of actions in each iteration (top); The sum of the rewards and value of ε in each iteration (bottom) of the ducking underneath an obstacle use case.

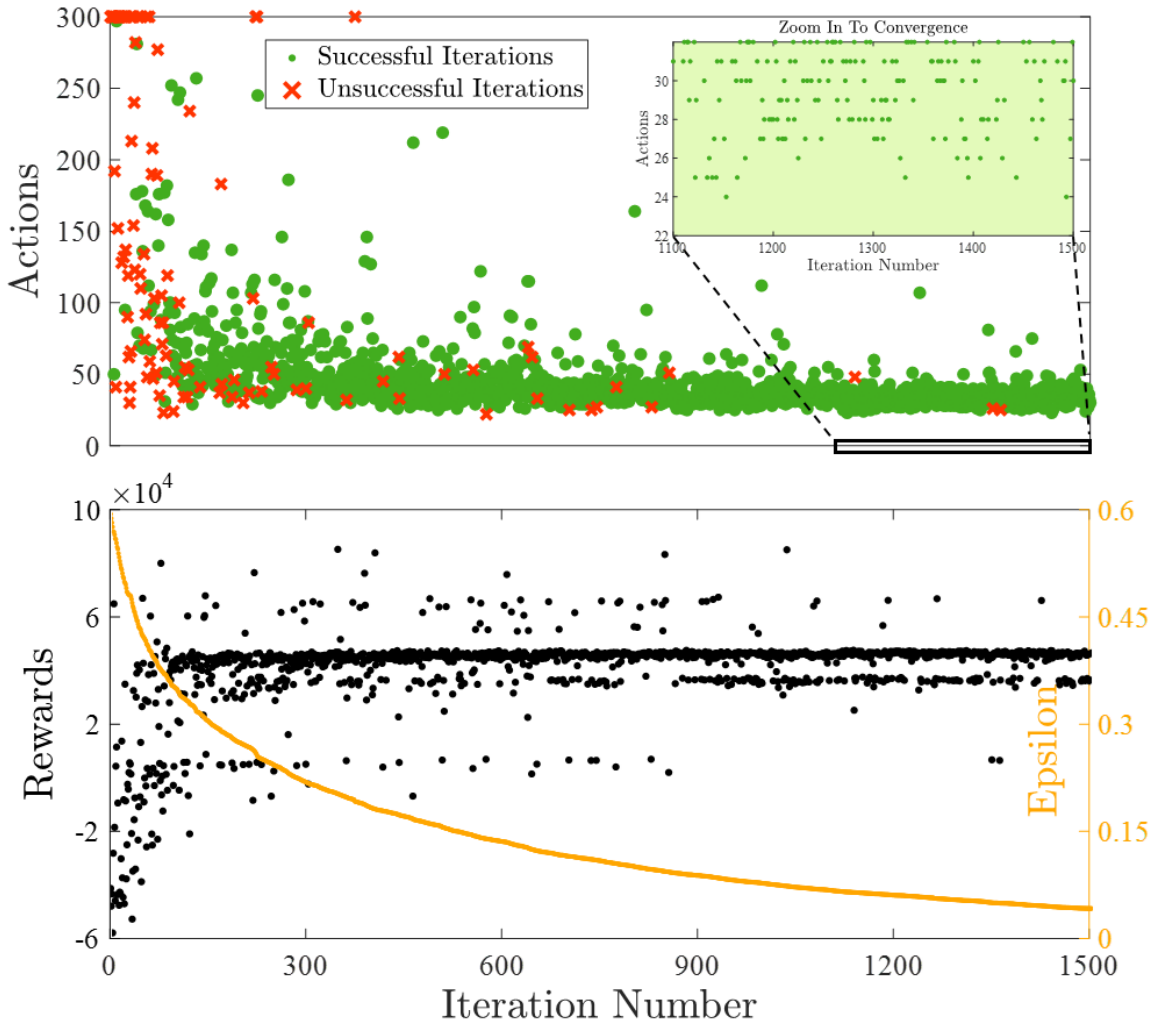


Figure 7.3 - The number of actions in each iteration (top); The sum of the rewards and value of ε in each iteration (bottom) of the climbing on top of an obstacle use case.

7.2 Squeezing Through a Channel

The simulation converged to a successful solution (with 17 actions) after about 600 iterations. The simulation presented in Figure 7.4 shows that starting in (a) for a channel width of 240 mm, the simulated robot performed four actions to lower its sprawl from 48 to 18 degrees and to decrease its width from 240 mm to 163 mm. When the width of the robot was smaller than that of the channel (180 mm), the simulated robot continued advancing to reach its target. (see video [48]).

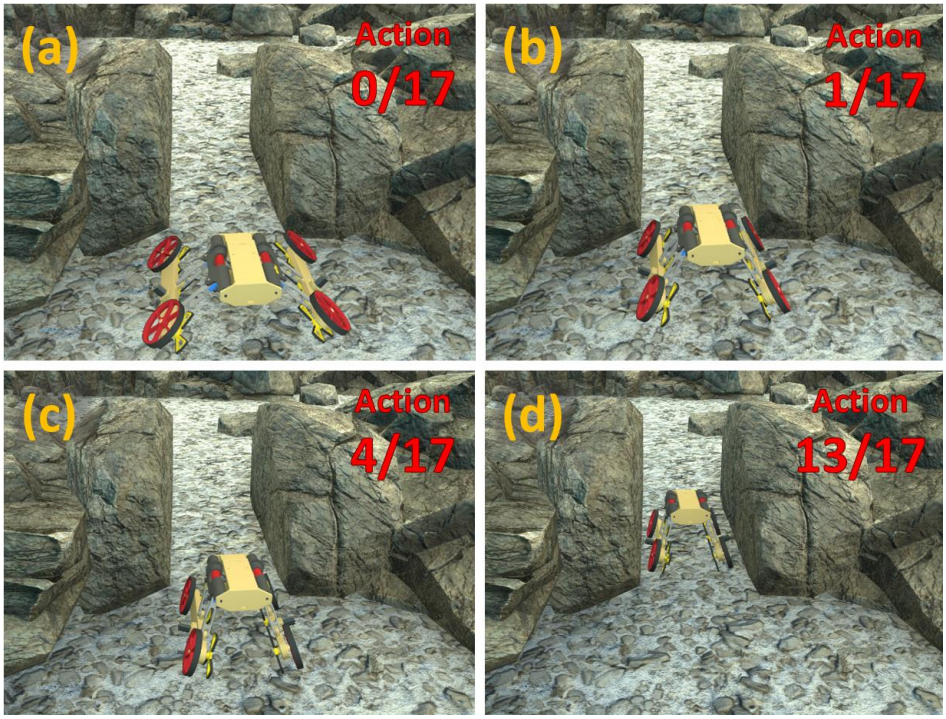


Figure 7.4 – The simulated RSTAR crawling between two walls. Starting at (a), the robot increased the sprawl to reduce its width (b-c) and continued advancing toward its target (see video [48]).

7.3 Crawling Underneath an Obstacle

In this use case the simulation converged after 700 iterations to a solution with 14 actions. Starting from an initital configuration in which the simulated RSTAR's height was 100 mm (compared to the 55 mm clearance of the obstacle), the simulated RSTAR first moved its FBEM forward (b) and then lowered its sprawl until reaching a height of 51 mm (c), then it continued to advance (d) to reach its target.

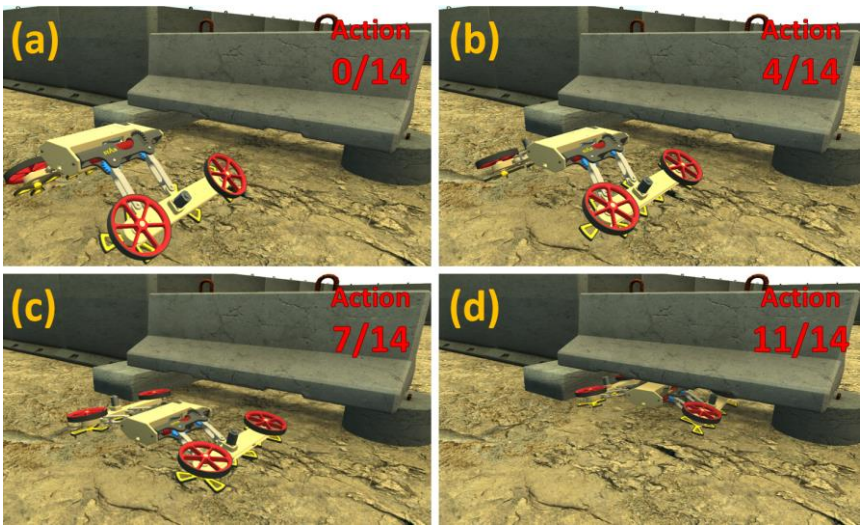


Figure 7.5 – Starting from its initial configuration (a), the simulated RSTAR moved its FBEM forward (b), then lowered it sprawl (c) and continued advancing forward. (see video [48]).

7.4 Climbing on Top of an Obstacle

The RSTAR was trained to climb from the ground as in Figure 7.6 (a) to the top of the step as in Figure 7.6 (f). Given the fact that the obstacle was substantially higher than the radius of the wheels, this use case was complex (even for experienced human operators). It required more complex dynamic maneuvers relative to the previous two use cases and its solution required more actions to perform. In order to reduce the learning time and ensure solution convergence,

the learning process was divided into two parts. In the first part, the agent was initially placed with its wheels on the tip of the step (Figure 7.6 (d)) and had to learn how to finish climbing as illustrated in Figure 7.6 (d)-(f). After the convergence of the first part, the Q values were saved and were set to be the initial values of the second learning part (the random action factor ϵ was reset to 0.6).

Thus, at the start of the second learning part, the robot already knew how to climb from d to f. In the second learning part, the agent learned how to start climbing from the ground (a) until it placed its wheels at the tip of the step (c). Together with the first learning part, the robot successfully learned how to fully climb over the step starting from the ground.

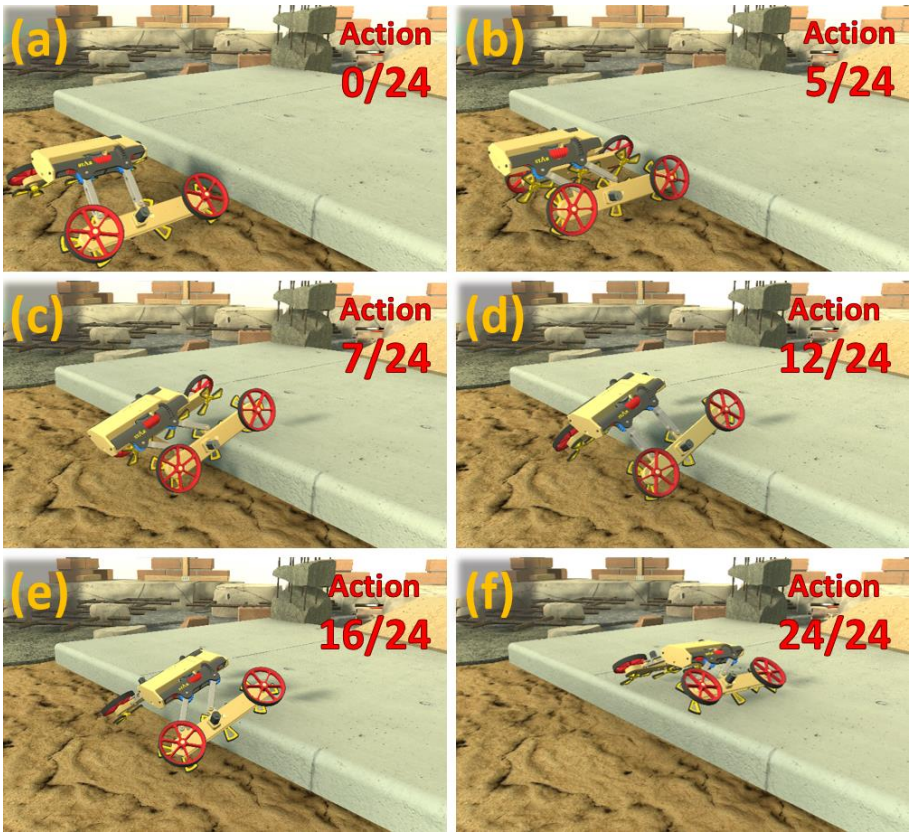


Figure 7.6 - The RSTAR climbing on top of an obstacle. Starting in (a), the robot moved its COM backward (b) to pitch upwards (c). Then it moved its COM forward (d) and lowered it (e) to finish climbing (f). (see video [48])

The algorithm produced multiple solutions to climb over the obstacle, with a range of 24 to 27 actions. The difference between the different solutions is analyzed in Section (IV.E).

Figure 7.6 illustrates how the simulated RSTAR climbed over a step obstacle. The step was 50 mm high compared to the 58 mm diameter of the RSTAR's wheels. Starting in (a), the simulated robot moved its COM backward using its FBEM (b) and advanced forward to pitch upwards and place its front wheels on the edge of the obstacle (c). Note e that dynamically, moving the COM was essential to allow pitching upwards. After placing its front wheels and leaning on the step, it sprawled down to lower it COM (d) and then advanced its COM forward using the FBEM (e). Both actions were also critical to climbing, since otherwise the robot would pitch on its back when it attempted to advance forward. The robot continued advancing using its wheels to finish climbing over the obstacle (f). (See video [48]).

7.5 Solution Suitability for Untrained Obstacle Sizes

To examine the level of compatibility of the solutions to obstacle size variation, we tested the learned solutions with obstacles of different sizes (Table 2). To squeeze through a channel, the solution learned for the 180 mm wide channel emerged as suitable for narrower channels up to 164 mm wide. For ducking underneath an obstacle, the solution for a clearance of 55 mm was appropriate for a 51 mm clearance. To climb over an obstacle, the solution with the smallest number of actions (24) was only suitable for climbing over the obstacle it was trained on (50 mm). However, solutions with more actions were more versatile. Two solutions with 26 and 27 actions allowed the robot to climb on top of obstacles that were 51 and even 55 mm high.

Table 2 - Compatibility of the simulation results.

Use Case	Number of Actions	Solution Compatibility
Squeezing through a 180 mm channel.	17	164 mm and wider
Ducking underneath a 55 mm opening.	14	51 mm and higher
Climbing over a 50 mm obstacle.	24	Up to 50 mm
	26	Up to 51 mm
	27	Up to 55 mm

7.6 Outperforming Human Experts

To estimate the quality of the learning process, we compared the algorithm's results to human solutions. This comparison was conducted for the most challenging case of climbing over a step. Human solutions were composed from two groups of six M.Sc. students from the Bio-Inspired and Medical Robotics Lab., who are familiar with the physical RSTAR robot and its kinematics. Both groups were asked to solve the climbing use case using the simulation. Each student was given an explanation of the different action possibilities and then was allotted one full hour to find a solution. The students worked separately and were not allowed to work with each other.

The first group composed a solution by listing a sequence of actions to perform, the students were told how many actions the simulation needed (24) but not the action set. The students in the second group controlled the RSTAR using a joystick and their performance was recorded in terms of motors actuation. The results of the second group are presented in Figure 7.7 - Figure 7.9.

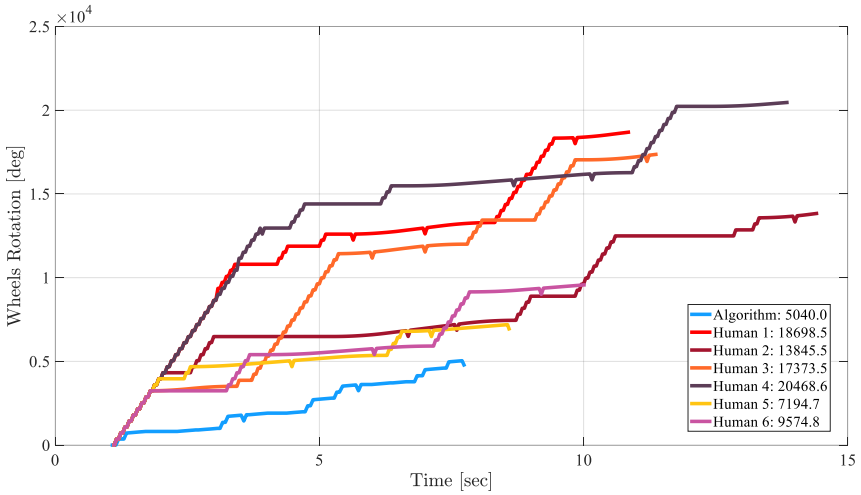


Figure 7.7 – Rotation of the RSTAR wheels in the solutions of the second group compared to the algorithm.

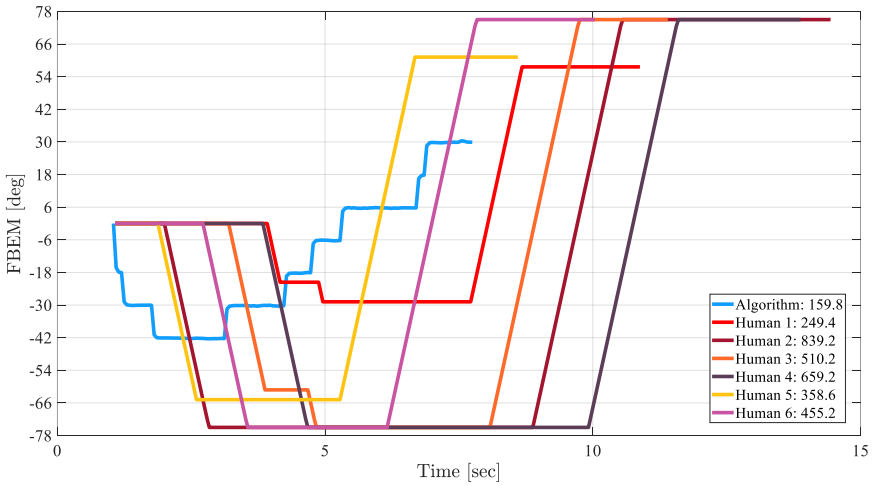


Figure 7.8 –The change of the FBEM angle in the solutions of the second group compared to the algorithm.

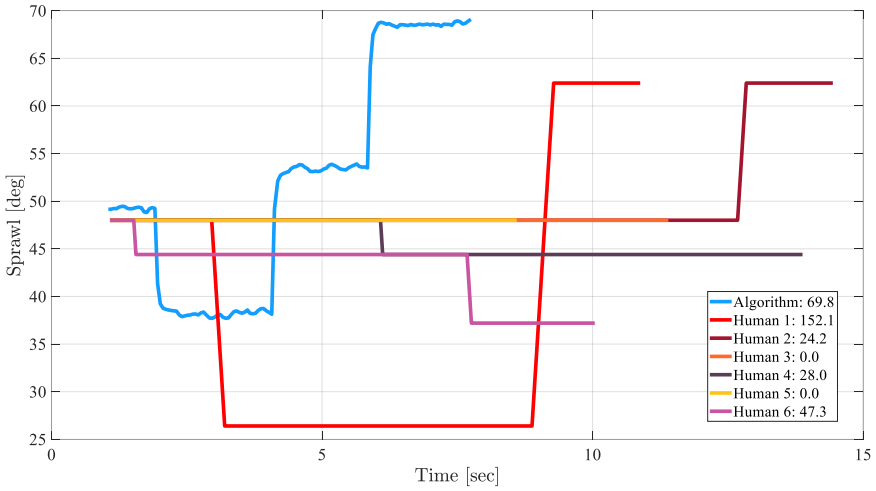


Figure 7.9- The change of the sprawl angle in the solutions of the second group compared to the algorithm

Interestingly, and surprisingly at least to the authors of this manuscript, the Q learning algorithm outperformed all the "human experts" in both groups. In the first group, none of the students was able to find a solution with a lesser number of actions. Two students were not able to find a climbing solution at all, while four others found solutions ranging from 25 to 28 actions compared to the Q learning solution of 24 actions. In the second group, all the students succeeded to climb over the step using the joystick and two students even succeeded to climb without changing the sprawl mechanism but their total solutions demanded more actuation of the FBEM and driving motors as presented in Figure 7.7- Figure 7.9. Table III summarize the results of the second group compared to the algorithm.

Table 3 - Comparison between the algorithm result to human solutions.

		Time [sec]	Sprawl [deg]	FBEM [deg]	Wheels Rotation [deg]
Algorithm		7.8	69.8	159.7	5,039
Humans	Average	11.5	41.9	511.9	14,525
	Standard deviation	2.2	56.9	211.8	5,283

8 Hardware Implementation and Validation

This section presents the implementation of algorithm's results on an actual RSTAR prototype. In each experiment, the initial conditions of the RSTAR (position and configuration) and the obstacle's geometric properties were (nearly) identical to the learned dimensions. In each case, a sequence of actions was uploaded to the RSTAR's control system. The uploaded sequence was the set learned using the Q learning algorithm. To squeeze through a channel the robot reduced its width and advanced between the two walls (Figure 8.1). The robot also successfully lowered its body and ducked underneath the obstacle (Figure 8.2). In both cases, the robot successfully performed its task in the physical environment as of the first attempt.

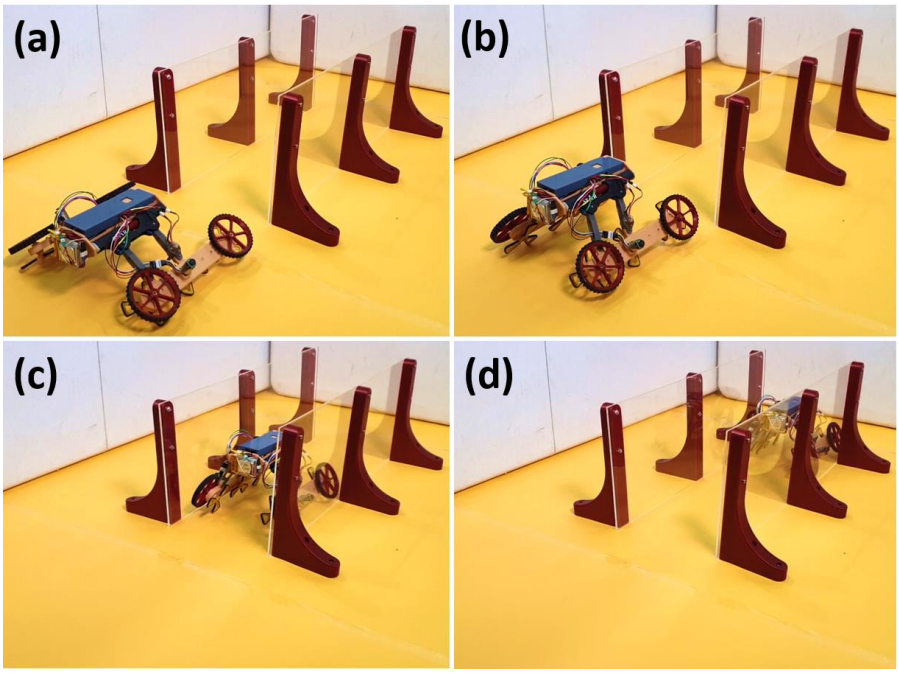


Figure 8.1 - RSTAR successfully reduced its width (b) and crawled between two walls 18 cm apart (c)-(d). (See video [48]).

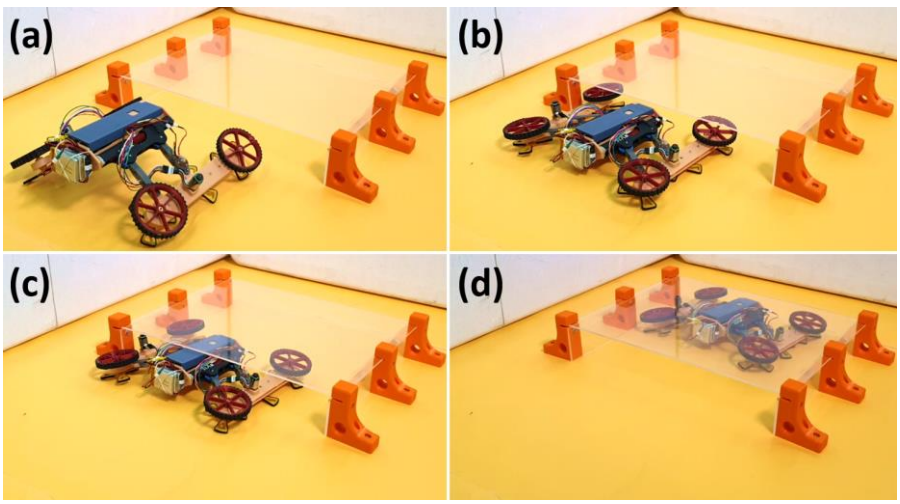


Figure 8.2 - RSTAR lowered its body and crawled underneath an obstacle 5.5 cm in height. (See video[48]).

The robot was able to climb over obstacles measuring 45 mm in height but not obstacles 50 mm high using its minimal 24 actions sequence. The 10% difference between the simulation and the actual robot can be attributed to the minor differences in the definition of the mechanical and geometrical properties and the accuracy of the solution of the physical engine. Interestingly, the robot successfully climbed over the 50 mm high obstacles using the 27 action solution (which according to simulation would allow it to climb over 55 cm steps). This result is consistent with the Q learning solution which suggested that the 27 actions solution would increase its climbing capability by 5 mm.

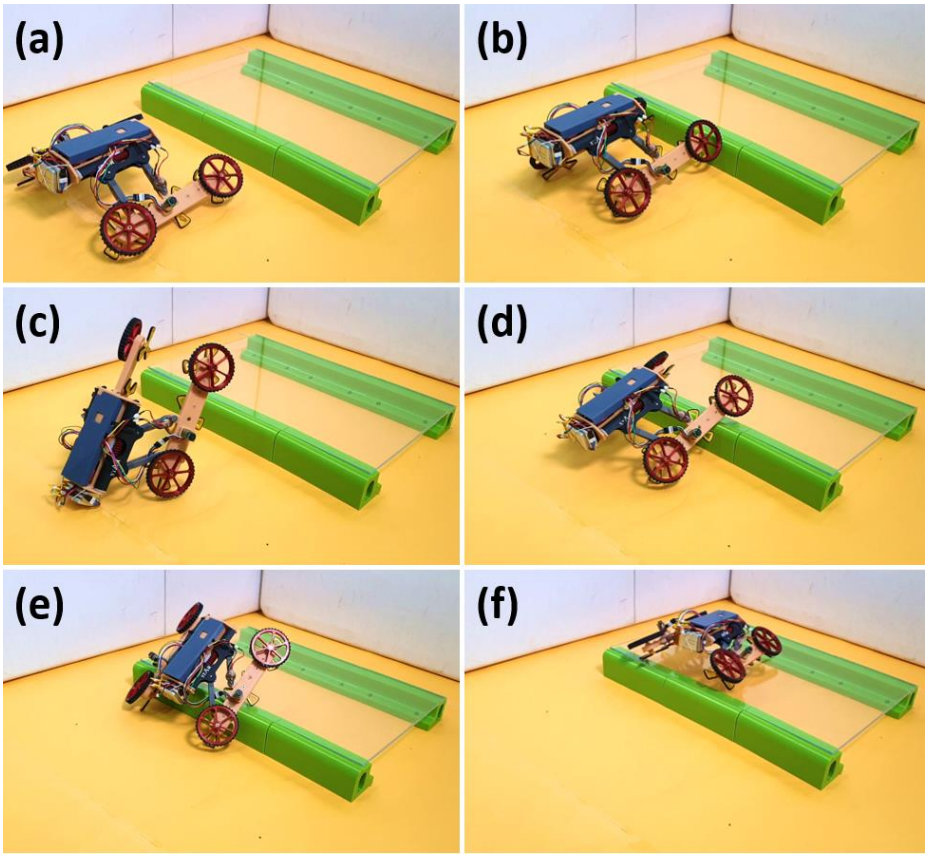


Figure 8.3 - The RSTAR successfully climbs over an obstacle 50 mm in height (27 actions solution).

9 Conclusions

In this thesis, we presented a novel sprawled tuned reconfigurable robot that can perform multiple maneuvers without any external modification. The robot is fitted with two unique mechanisms; a sprawl mechanism that tilts the rotation axis of the legs, and a four bar extension mechanism (FBEM) that prolongs the distance between the body and the legs and moves the COM in the fore-aft direction. The sprawl allows the robot to change its dynamics from the lateral to the sagittal plane and for inverted locomotion. The FBEM extends the length of the legs while keeping them parallel to the body. Using a combination of these two mechanisms, the robot can change its width and height three fold (and even more if we use longer extension bars) and move the COM in both the fore-aft and vertical directions.

The RSTAR can perform multiple original locomotion movements and execute many climbing maneuvers thus outperforming our previously designed STAR robot. The RSTAR can perform a unique turtle locomotion gait which allows the robot to crawl over extremely soft surfaces such as thick mud or sand where the wheels would get entrenched. The turtle gait can also be used to climb over obstacles whose height is greater than the diameter of its spoke wheels. By extending/narrowing its width the robot is capable of crawling vertically in a tube or a canal by applying pressure to the walls.

The RSTAR can also flip its body upside down by changing the position of its COM. This feature can be used to decrease its cost of transport and reduce oscillations by fitting its legs with regular wheels on one side for running over smooth surfaces and fitting spoke legs on the other side for running over unstructured terrains.

In addition to the mechanical design, we also applied a *Q* learning algorithm in order to learn to overcome autonomously three typical obstacle use cases. These included squeezing through a channel, ducking underneath an obstacle, and climbing over an obstacle (which requires a very skilled operator). To simplify the process, safeguard the robot and reduce the learning time, all the learning took place in a simulated environment using Unity software. The algorithm found solutions to all the use cases.

Results were implemented on the physical RSTAR. The robot was placed in an environment reminiscent of the simulated environment and preprogrammed to perform the learned set of actions. With no human intervention, the RSTAR successfully overcame all three obstacles on its first attempts.

The learned solutions are also suitable for similar obstacles with different geometries. For instance, the solution of squeezing through a channel was developed for a width of 180 mm but is also suitable for a narrower channel 164 mm wide. The same sequence for climbing over a 50 mm obstacle is also compatible for steps with lower heights. To climb over an obstacle, the algorithm provided multiple solutions with different numbers of actions. An analysis of the solutions showed that some of the solutions with a larger number of actions were suitable for climbing over higher obstacles. For example, the simulated robot climbed over a 50 mm step with 24 actions, and over a 55 mm step with 27 actions. This result was also validated in experiments.

The climbing use case was especially challenging. Although the algorithm had no prior knowledge of the kinematics of RSTAR or the obstacles, it generated mechanically intelligent results for climbing on the step. The result included moving the COM backward and then forward and changing its height to overcome the obstacle. This solution outperformed other solutions devised by two groups of human experts.

Based on these encouraging results, future work will focus on the inclusion of perception capabilities in the learning setup to learn action sequences directly based on perceptual input.

10 References

[1] J. M. Morrey, B. Lambrecht, A. D. Horchler, R. E. Ritzmann, and R. D. Quinn, "Highly mobile and robust small quadruped robots", IEEE Int. Conf. on Intelligent Robots and Systems, Vol. 1, pp. 82-87, 2003.

[2] M. Hoover, S. Burden, X. Y. Fu, S. S. Sastry, and R. S. Fearing, "Bio-inspired design and dynamic maneuverability of a minimally actuated six-legged robot", IEEE Int. Conf. on Biomedical Robotics and Biomechatronics, pp. 869-873, 2010.

[3] P. Birkmeyer, K. Peterson, and R. S. Fearing, "DASH: A dynamic 16g hexapedal robot", IEEE Int. Conf. on Intelligent Robots and Systems, pp. 2683-2689, 2009.

[4] S. Kim, J. E. Clark, and M. R. Cutkosky, "iSprawl: Design and turning of high-speed autonomous open-loop running", The Int. Journal of Robotic Research, Vol. 25, No.9. pp. 903-912, 2006.

[5] O. Pullin, N. J. Kohut, D. Zarrouk, and R.S. Fearing, "Dynamic Turning of 13cm robot comparing tail and differential drive", IEEE Int. Conf. on Robotics and Automation, pp. 5083-5093, 2012.

[6] K. C. Galloway, G. C. Haynes, B. D. Ilhan, A. M. Johnson. R. Knopf, G. Lynch, B. Plotnick, M. White, and D. E. Koditschek, "X-RHex: a highly mobile hexapedal robot for sensorimotor tasks", University of Pennsylvania Technical Report, 2010.

[7] J. G. Cham, S. A. Bailey, J. E. Clark, R. J. Full, and M. R. Cutkosky, "Fast and robust: Hexapedal robots via shape deposition manufacturing", The International Journal of Robotics Research, Vol. 21, No. 10-11, pp. 869-882, 2002.

[8] D. Zarrouk, A. Pullin, N. J. Kohut, and R. S. Fearing, "STAR - Sprawl Tuned Autonomous Robot", IEEE Int. Conf. on Robotics and Automation, pp. 20-25, 2013.

[9] D. Zarrouk, and R. S. Fearing, "Controlled In-Plane Locomotion of a Hexapod Using a Single Actuator", IEEE Trans. on Robotics, Vol. 31, No. 1, pp. 157-167, 2015.

[10] P. K. Karidis ,D. Zarrouk, I. Poulakakis, R. S. Fearing, and H.G. Tanner, "Planning with the STAR(s)", IEEE Int. Conf. on Intelligent Robots and Systems, pp. 3033-3038, 2014.

[11] T. M. Kubow, and R. J. Full, "The role of the mechanical system in control: A hypothesis of self stabilization in hexapedal runners", *Philosophical Transactions: Biological sciences*, Vol. 354, pp. 849-861, 1999.

[12] D. I. Jindrich, and R. J. Full, "Dynamic stabilization of rapid hexapedal locomotion", *Journal of Experimental Biology*, Vol. 205, pp. 2803-2823, 2002.

[13] H. Komsuoglu, K. Sohn, R. J. Full, and D. E. Koditschek, "A physical model for dynamical arthropod running on level ground", 11th ISER, 2008.

[14] J. E. Seipel, and P. Holmes, "Running in three dimensions: analysis of a point-mass sprung-leg model". *The International Journal of Robotics Research*, Vol. 24, No. 8, pp. 677-674, 2005.

[15] J. Seipel, and P. Holmes, "A simple model for clock-actuated legged locomotion", *Regular and Chaotic Dynamics*, Vol. 12. No. 5. pp. 502-520, 2007.

[16] R. P. Kukillaya, and P. Holmes, "A hexapedal jointed-leg model for insect locomotion in the horizontal plane", *Biol. Cyber.*, DOI 10.1007/s00422-007-0180-2, 2007.

[17] J. Schmitt, and P. Holmes, "Mechanical models for insect locomotion: dynamics and stability in the horizontal plane I. Theory", *Biol. Cyber.* Vol. 83, pp. 501-515, 2000.

[18] J. Schmitt, and P. Holmes, "Mechanical models for insect locomotion: dynamics and stability in the horizontal plane II. Application", *Biol. Cyber.* Vol. 83, pp. 517-527, 2000.

[19] J. E. Seipel, P. J. Holmes, and R. J. Full, "Dynamics and stability of insect locomotion: a hexapedal model for horizontal plane motions", *Biol. Cyber.* Vol. 91, pp. 6-90, 2004.

[20] Y. S. Kim, G. P. Jung, H. Kim, K. J. Cho, and C. N. Chu, "Wheel Transformer: A Wheel-Leg Hybrid Robot With Passive Transformable Wheels", IEEE trans. On Robotics, Vol. 30, N6, pp. 1487-1498, 2014.

[21] M. J. Spenko, G. C. Haynes, J. A. Saunders, M. R. Cutkosky, A. A Rizzi, R. J. Full, and D. E. Koditschek, "Biologically Inspired Climbing with a Hexapedal Robot," *Journal of Field Robotics*, Vol. 25, No. 4, 2008.

[22] J. T. Karras, C. L. Fuller, K. C. Carpenter, A. Buscicchio, D. McKeeby, C. J. Norman, C.E. Parcheta, I. Davydychev, and R. S. Fearing, Pop-up Mars Rover with Textile-Enhanced Rigid-Flex PCB Body, IEEE Int. Conf. Robotics and Automation, Singapore, May 2017.

[23] P. K. Karidis ,D. Zarrouk, I. Poulakakis, R. S. Fearing, and H.G. Tanner, "Planning with the STAR(s)", IEEE Int. Conf. on Intelligent Robots and Systems, pp. 3033-3038, 2014.

[24] Youtube video, Rising STAR, a miniautre highly reconfigurable robot: https://youtu.be/XI_aepVAuxY
[25] T. N. Mazouchova, P. B. Umbanhowar, and D. I. Goldman, “Flipper-driven terrestrial locomotion of a sea turtle-inspired robot”, *Bioinspiration and Biomimetics*, Vol. 8, No. 2, 2013.
[26] D. Zarrouk, and M. Shoham, “Analysis and design of one degree of freedom worm robots for locomotion on rigid and compliant terrain”, ASME, *Journal of Mechanisms and Robotics*, Vol. 134, No. 2, 2012.
[27] R. S. Sutton and A. G. Barto. Reinforcement Learning: An Introduction. Second Edition, MIT Press, 2012.
[28] Kober J., Peters J. (2014) Reinforcement Learning in Robotics: A Survey. In: Learning Motor Skills. Springer Tracts in Advanced Robotics, vol 97. Springer, Cham.
[29] C. J. C. H. Watkins and P. Dayan, “Learning from delayed rewards,” Ph.D. dissertation, University of Cambridge, Cambridge, UK, 1989.
[30] Honglak Lee, Yirong Shen, Chih-Han Yu, G. Singh and A. Y. Ng, "Quadruped robot obstacle negotiation via reinforcement learning," Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006., Orlando, FL, 2006, pp. 3003-3010.
[31] K. Zhang, F. Niroui, M. Ficocelli and G. Nejat, "Robot Navigation of Environments with Unknown Rough Terrain Using deep Reinforcement Learning," 2018 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR), Philadelphia, PA, 2018, pp. 1-7.
[32] F. Niroui, K. Zhang, Z. Kashino and G. Nejat, "Deep Reinforcement Learning Robot for Search and Rescue Applications: Exploration in Unknown Cluttered Environments," in IEEE Robotics and Automation Letters, vol. 4, no. 2, pp. 610-617, April 2019.
[33] F. Niroui, B. Sprenger and G. Nejat, "Robot exploration in unknown cluttered environments when dealing with uncertainty," 2017 IEEE International Symposium on Robotics and Intelligent Sensors (IRIS), Ottawa, ON, 2017, pp. 224-229.
[34] V. M. Babu, U. V. Krishna and S. K. Shahensha, "An autonomous path finding robot using Q-learning," 2016 10th International Conference on Intelligent Systems and Control (ISCO), Coimbatore, 2016, pp. 1-6.
[35] M. Totani, N. Sato and Y. Morita, "Step climbing method for crawler type rescue robot using reinforcement learning with Proximal Policy Optimization," 2019 12th International Workshop on Robot Motion and Control (RoMoCo), Poznań, Poland, 2019, pp. 154-159.
[36] A. Dutta, P. Dasgupta and C. Nelson, "Adaptive locomotion learning in modular self-reconfigurable robots: A game theoretic approach," 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vancouver, BC, 2017, pp. 3556-3561.
[37] G. Shi, et al. "Neural lander: Stable drone landing control using learned dynamics." 2019 International Conference on Robotics and Automation (ICRA). IEEE, 2019.
[38] A. Nagabandi et al. "Neural network dynamics models for control of under-actuated legged millirobots." arXiv preprint arXiv:1711.05253 (2017).
[39] J. Tan, et al. "Sim-to-real: Learning agile locomotion for quadruped robots." arXiv preprint arXiv:1804.10332 (2018).
[40] T.P. Lillicrap, et al. "Continuous control with deep reinforcement learning." arXiv preprint arXiv:1509.02971 (2015).
[41] Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. The MIT Press, Cambridge (1998).
[42] Watkins, C., Dayan, P.: Technical note: Q-learning. Machine Learning 8 (1992), 279–292.
[43] Watkins, C.: Learning from Delayed Rewards. PhD thesis, Cambridge University, Cambridge, England (1989).
[44] Matignon, Laëtitia & Laurent, Guillaume & Fort-Piat, Nadine. (2006). Reward Function and Initial Values: Better Choices for Accelerated Goal-Directed Reinforcement Learning. Lecture Notes in Computer Science.
[45] Mataric, M.J.: Reward functions for accelerated learning. In: Proc. of the 11th ICML. (1994) 181–189.
[46] Ng, A.Y., Harada, D., Russell, S.: Policy invariance under reward transformations: theory and application to reward shaping. In: Proc. of the 16th ICML. (1999), 278–287.
[47] Web link address: https://unity.com/
[48] Youtube video Overcoming Obstacles Using Reinforcement Learning in RSTAR Robot: https://www.youtube.com/watch?v=oaCa737z7uE

11 Appendices

[A] 3D Printer - Objet Connex3

http://www.stratasys.com/3d-printers/objet-350-500-connex3

Objet350 and Objet500 Connex3

Driven by powerful PolyJet™ technology

Proven PolyJet 3D Printing is famous for smooth surfaces, fine precision and diverse material properties. It works a bit like inkjet document printing, but instead of jetting drops of ink onto paper, the print head jets microscopic layers of liquid photopolymer onto a build tray and instantly cures them with UV light. The fine layers build up to create a prototype or production part.

Along with the selected model material, the 3D printer features two support material options: SUP705, removed with a WaterJet; and SUP706, which is easily removed and soluble for automated post-processing and increased geometric freedom to print complex and delicate features and small cavities.

With its astonishingly realistic aesthetics and ability to deliver special properties such as transparency, flexibility and even bio-compatibility, PolyJet 3D Printing offers a competitive edge in consumer products prototyping, precision tooling and specialized production parts.

System Specifications	
Model Materials	Rigid Opaque: VeroPureWhite™, VeroWhitePlus™, VeroBlackPlus™, VeroGray™, VeroBlue™, VeroCyan™, VeroMagenta™, VeroYellow™ Rubber-like: TangoPlus™, TangoBlackPlus™, TangoBlack™, TangoGray™ Transparent: VeroClear™ and RGD720 Simulated Polypropylene: Rigur™ and Durus™ High Temperature Bio-compatible
Digital Materials	Digital ABS™ and Digital ABS2™ in ivory and green Hundreds of vibrant, repeatable colors in opaque and translucent Rubber-like blends in a range of Shore A values and color Simulated polypropylene materials with improved heat resistance
Material Options	Over 1,000
Maximum Materials per Part	82
Support Material	SUP705 (WaterJet removable)
	SUP706 (soluble)
Maximum Build Size (XYZ)	Objet350: 342 x 342 x 200 mm (13.4 x 13.4 x 7.9 in.) Objet500: 490 x 390 x 200 mm (19.3 x 15.4 x 7.9 in.)
System Size and Weight	1400 x 1260 x 1100 mm (55.1 x 49.6 x 43.4 in.); 430 kg (948 lbs.) *Material Cabinet:* 330 x 1170 x 640 mm (13 x 46.1 x 26.2 in.); 76 kg (168 lbs.)
Resolution	X-axis: 600 dpi; Y-axis: 600 dpi; Z-axis: 1600 dpi
Accuracy	20-85 microns for features below 50 mm; up to 200 microns for full model size
Minimum Layer Thickness	Horizontal build layers as fine as 16 microns (.0006 in.)
Build Modes	Digital Material: 30-micron (.001 in.) resolution High Quality: 16-micron (.0006 in.) resolution High Speed: 30-micron (.001 in.) resolution
Software	Objet Studio intuitive 3D printing software
Workstation Compatibility	Windows 7/ Windows 8
Network Connectivity	LAN - TCP/IP
Operating Conditions	Temperature 18-25°C (64-77°F); relative humidity 30-70% (non-condensing)
Power Requirements	110-240 VAC 50/60Hz; 1.5 kW single phase
Regulatory Compliance	CE, FCC

[B] Printing Material – VeroWhitePlus

https://store.stratasys.com/stratasysstorefront/stratasys/en/Materials-%26-Service-Consumables/NA/Flavors/VeroWhitePlus%2C-RGD835/p/P034

POLYJET MATERIALS DATASHEET

RIGID OPAQUE MATERIALS

VERO PUREWHITE RGD837, VEROGRAY RGD850, VEROBLACKPLUS RGD875, VEROWHITEPLUS RGD835, VEROYELLOW RGD836, VEROCYAN RGD841, VEROMAGENTA RGD851

	ASTM	UNITS	METRIC	UNITS	IMPERIAL
Tensile strength	D-638-03	MPa	50-65	psi	7250-9450
Elongation at break	D-638-05	%	10-25	%	10-25
Modulus of elasticity	D-638-04	MPa	2000-3000	psi	290,000-435,000
Flexural Strength	D-790-03	MPa	75-110	psi	11000-16000
Flexural Modulus	D-790-04	MPa	2200-3200	psi	320,000-465,000
HDT, °C @ 0.45MPa	D-648-06	°C	45-50	°F	113-122
HDT, °C @ 1.82MPa	D-648-07	°C	45-50	°F	113-122
Izod Notched Impact	D-256-06	J/m	20-30	ft lb/inch	0.375-0.562
Water Absorption	D-570-98 24hr	%	1.1-1.5	%	1.1-1.5
Tg	DMA, E«	°C	52-54	°F	126-129
Shore Hardness (D)	Scale D	Scale D	83-86	Scale D	83-86
Rockwell Hardness	Scale M	Scale M	73-76	Scale M	73-76
Polymerized density	D792	g/cm3	1.17-1.18		
Ash content VeroGray, VeroWhitePlus	USP281	%	0.23-0.26	%	0.23-0.26
Ash content VeroBlackPlus	USP281	%	0.01-0.02	%	0.01-0.02

[C] 3D Printer - FORMLABS FORM 2

https://formlabs.com/3d-printers/form-2/

Technical Specifications

PRINTER

Price	$3499
Dimensions	35 × 33 × 52 cm 13.5 × 13 × 20.5 in
Weight	13 kg / 28.5 lbs
Operating Temperature	Autoheats to 35° C or 95° F Self-heating Resin Tank
Power Requirements	100–240 V 1.5 A 50/60 Hz 65 W
Laser Specifications	EN 60825-1:2007 certified Class 1 laser product 405 nm violet laser 250 mW laser
Connectivity	Wi-Fi, Ethernet, and USB
Printer Control	Interactive touch screen

PREFORM SOFTWARE

System Requirements	Windows 7 and up Mac OS X 10.7 and up
File Type	.STL or .OBJ

PRINTING PROPERTIES

Technology	Stereolithography (SLA)
Peel Mechanism	Sliding peel process with wiper
Resin Fill System	Automated cartridge system
Build Volume	145 × 145 × 175 mm 5.7 × 5.7 × 6.9 in
Layer Thickness (Axis Resolution)	25, 50, 100, 200 microns 0.001, 0.002, 0.004, 0.008 in.
Laser Spot Size (FWHM)	140 microns 0.0055 inches
Supports	Auto-generated Easily removable

FINISHING KIT

Includes

- Finishing tray
- Scraper
- Pre and post-rinse tubs
- Rinse basket
- Squeeze bottle
- Flush cutters
- Tweezers
- Disposable Nitrile gloves
- Removal tool
- Removal jig

Contact

ORDER TODAY
formlabs.com/store

LEARN MORE
formlabs.com/dentistry

QUESTIONS?
hello@formlabs.com
+1 617 702 8476

[D] Printing Material - Form Resins

https://archive-media.formlabs.com/upload/XL-DataSheet.pdf

STANDARD RESINS

CLEAR FLGPCL04 | WHITE FLGPWH04 | GREY FLGPGR04 | BLACK FLGPBK04 | COLOR BASE FLGPCB01

	METRIC[1]		IMPERIAL[1]		METHOD
	Green[2]	Post-Cured[3]	Green[2]	Post-Cured[3]	
Tensile Properties					
Ultimate Tensile Strength	38 MPa	65 MPa	5510 psi	9380 psi	ASTM D 638-10
Tensile Modulus	1.6 GPa	2.8 GPa	234 ksi	402 ksi	ASTM D 638-10
Elongation at Break	12 %	6 %	12 %	6 %	ASTM D 638-10
Flexural Properties					
Flexural Modulus	1.3 GPa	2.2 GPa	181 ksi	0.5 ksi	ASTM C 790-10
Impact Properties					
Notched IZOD	16 J/m	25 J/m	0.3 ft-lbf/in	0.46 ft-lbf/in	ASTM D 256-10
Thermal Properties					
Heat Deflection Temp. @ 1.8 MPa	42.7 °C	58.4 °C	108.9 °F	137.1 °F	ASTM D 648-07
Heat Deflection Temp. @ 0.45 MPa	49.7 °C	73.1 °C	121.5 °F	163.6 °F	ASTM D 648-07

DENTAL MODEL

FLDMBE02

	METRIC[1]		IMPERIAL[1]		METHOD
	Green[2]	Post-Cured[3]	Green[2]	Post-Cured[3]	
Tensile Properties					
Tensile Strength at Yield	33 MPa	61 MPa	4800 psi	8820 psi	ASTM D 638-14
Tensile Modulus	1.0 GPa	2.7 GPa	230 ksi	397 ksi	ASTM D 638-14
Elongation at Failure	25 %	5 %	25 %	5 %	ASTM D 638-14
Flexural Properties					
Flexural Modulus	0.95 GPa	2.5 GPa	138 ksi	365 ksi	ASTM D 790-15
Flexural Strength at 5% Strain	33.9 MPa	95.8 MPa	4910 psi	13900 psi	ASTM D 790-15
Impact Properties					
Notched IZOD	27 J/m	33 J/m	0.5 ft-lbf/in	0.6 ft-lbf/in	ASTM D256-10
Thermal Properties					
Heat Deflection Temp. @ 1.8 MPa	32.8 °C	45.9 °C	91.1 °F	114.6 °F	ASTM D 648-16
Heat Deflection Temp. @ 0.45 MPa	40.4 °C	48.5 °C	104.7 °F	119.3 °F	ASTM D 648-16

GREY PRO RESIN

FLPRGR01

	METRIC[1]		IMPERIAL[1]		METHOD
	Green[2]	Post-Cured[3]	Green[2]	Post-Cured[3]	
Tensile Properties					
Ultimate Tensile Strength	33 MPa	61 MPa	5076 psi	8876 psi	ASTM D 638-14
Tensile Modulus	1.4 GPa	2.6 GPa	203 ksi	377 ksi	ASTM D 638-14
Elongation at Break	33 %	13 %	33 %	13 %	ASTM D 638-14
Flexural Properties					
Flexural Stress at 5% Strain	39 MPa	86 MPa	5598 psi	12400 psi	ASTM D 790-15
Flexural Modulus	0.9 GPa	2.2 GPa	136 ksi	319 ksi	ASTM D 790-15
Impact Properties					
Notched IZOD	not tested	18.7 J/m	not tested	0.351 ft-lbf/in	ASTM D256-10
Thermal Properties					
Heat Deflection Temp. @ 1.8 MPa	not tested	62.4 °C	not tested	144.3 °F	ASTM D 648-16
Heat Deflection Temp. @ 0.45 MPa	not tested	77.5 °C	not tested	171.5 °F	ASTM D 648-16
Coefficient of Thermal Expansion (-30 to 30° C)	not tested	78.5 µm/m/°C	not tested	43.4 µin/in/°F	ASTM E 831-13

[E] 3D Printer – UPBOX

https://www.tiertime.com/up-box-plus/

UP BOX Specifications

Printing Technology	MEM (Melted Extrusion Modeling)
Build Volume	255 x 205 x 205mm (W x H x D) 10" x 8" x 8"
Print Head	Single, Modular for easy replacement.
Z-Resloution	0.1/0.15/0.20 /0.25 /0.30 /0.35 /0.40 mm
Supporting Structure	Smart Support Technology: automactically generated, easy to remove, fine-tunable.
Platform Leveling	Fully automatic leveling with integrated leveling probe.
Print Surface	Heated bed with perf board
Unterthered Printing	Yes
Average Operational Noise	51dB
Advanced Features	Door Sensor, Air Filtration, Full-Color LED Bar
Bundled Software	UP Software
Compatible File Formats	STL, UP3, UPP
Connectivity	USB
Operating System	WinXP/Vista/7/8, Mac OS
Power adapter	110-240VAC, 50-60Hz, 220W
Chassis	Plastic case with metal frame, enclosed
Printer Weight	20KG / 44 LB
Printer Dimiension	493 x 493 x 517 mm (L x W x H) 19.5" x 19.5" x 20.5"
Weight with Packaging	30 Kg
Packaging Dimension	590 x 590 x 650mm (L x W x H); 22.4" x 22.4"x 24.8"

[F] 3D Printer – Ultimaker 2

https://ultimaker.com/download/7385/UserManual-UM2-v2.1.pdf

SPECIFICATIONS

Printing		
Print technology	Fused Filament Fabrication	
Build volume	223 mm / 223 mm / 205 mm	
Layer resolution	Fast:	200 micron (0.2 mm)
	Normal:	100 micron (0.1 mm)
	High:	60 micron (0.06 mm)
	Ulti:	40 micron (0.04 mm)
Positioning precision	12.5 micron / 12.5 micron / 5 micron	
Filament diameter	2.85 mm	
Nozzle diameter	0.4 mm	
Print speed	30 mm/s - 300 mm/s	
Travel speed	30 mm/s - 350 mm/s	
Software		
Supplied software	Cura - Official Ultimaker Software	
File types	STL / OBJ / DAE / AMF	
Supported OS	Windows / Mac / Linux	
Electrical		
AC Input	100 - 240 V	
	Approx. 1.4 AMPS	
	50 - 60 Hz	
	221 Watt max.	
Connectivity	Stand-alone SD card printing	
Physical dimensions		
Desktop space	357 mm / 342 mm / 388 mm	
Shipping dimensions	400 mm / 400 mm / 550 mm	
Weight	11.2 kg	
Shipping weight	18.0 kg	
Temperature		
Ambient operation temperature	15 - 32 °C	
Storage temperature	0 - 32 °C	
Nozzle operation temperature	180 - 260 °C	
Heated bed operation temperature	50 - 100 °C	
Sound		
Average operational noise	49 dBA	

CAUTION: The Ultimaker 2 generates high temperatures and has hot moving parts that can cause injury. Never reach inside of the Ultimaker 2 while it is in operation. Always control the Ultimaker 2 with the button at the front or with the power switch at the back. Allow the Ultimaker 2 to cool down for 5 minutes before reaching inside.

CAUTION: When opening the Ultimaker 2 for service, ensure that the power supply is turned off and the power cable is disconnected from the wall socket.

CAUTION: Only use the power supply that came with your Ultimaker 2.

[G] Printing Material – PLA

file:///C:/Users/%D7%9C%D7%99%D7%A8%D7%9F/Downloads/MSS_FDM_PLA_0118a.pdf

PLA

ECONOMY THERMOPLASTIC FOR STRATASYS F123 SERIES PRINTERS

At the core:
Advanced FDM Technology
Stratasys' FDM® (fused deposition modeling) technology works with engineering-grade thermoplastics to build strong, long-lasting and dimensionally stable parts with the best accuracy and repeatability of any FDM technology. These parts are tough enough to be used as advanced conceptual models, functional prototypes, manufacturing tools and production parts.

Meet production demands
FDM systems are as versatile and durable as the parts they produce. Advanced FDM 3D Printers boast the largest build envelopes and material capacities in their class, delivering longer, uninterrupted build times, bigger parts and higher quantities than other additive manufacturing systems, delivering high throughput, duty cycles and utilization rates.

Opening the way for new possibilities
FDM 3D Printers streamline processes from design through manufacturing, reducing costs and eliminating traditional barriers along the way. Industries can cut lead times and costs, products turn out better and get to market faster.

No special facilities needed
FDM 3D Printers are easy to operate and maintain compared to other additive fabrication systems because there are no messy powders or resins to handle and contain, and no special venting is required because FDM systems don't produce noxious fumes, chemicals or waste.

stratasys

MECHANICAL PROPERTIES[1]	TEST METHOD	ENGLISH		METRIC	
		XZ AXIS	*ZX AXIS*	*XZ axis*	*ZX axis*
Tensile Strength, Yield (Type 1, 0.125", 0.2"/min)	ASTM D638	6,580 psi	3,790 psi	45 MPa	26 MPa
Tensile Strength, Ultimate (Type 1, 0.125", 0.2"/min)	ASTM D638	6,990 psi	3,830 psi	48 MPa	26 MPa
Tensile Modulus (Type 1, 0.125", 0.2"/min)	ASTM D638	440,730 psi	368,200 psi	3,039 MPa	2,539 MPa
Elongation at Break (Type 1, 0.125", 0.2"/min)	ASTM D638	2.5%	1.0%	2.5%	1.0%
Elongation at Yield (Type 1, 0.125", 0.2"/min)	ASTM D638	1.5%	1.0%	1.5%	1.0%
Flexural Strength (Method 1, 0.05"/min)	ASTM D790	12,190 psi	6,570 psi	84 MPa	45 MPa
Flexural Modulus (Method 1, 0.05"/min)	ASTM D790	425,010 psi	358,290 psi	2,930 MPa	2,470 MPa
Flexural Strain at Break	ASTM D790	4.1%	1.9%	4.1%	1.9%
IZOD impact - notched (Method A, 23 °C)	ASTM D256	0.5 ft-lb/in	N/A	27 J/m	N/A
IZOD impact - unnotched (Method A, 23 °C)	ASTM D256	3.6 ft-lb/in	N/A	192 J/m	N/A

THERMAL PROPERTIES	TEST METHOD	ENGLISH	METRIC
Heat Deflection (HDT) @ 66 psi	ASTM D648	127 °F	53 °C
Heat Deflection (HDT) @ 264 psi	ASTM D648	124 °F	51 °C
Vicat Softening Temperature (Rate B/50)	ASTM D1525	129 °F	54 °C
Glass Transition Temperature (Tg)	DMA (SSYS)	145 °F	63 °C
Coefficient of Thermal Expansion (flow)	ASTM E831	$56x10^{-06}$ µin/(in·°F)	$101x10^{-06}$ µm/(m·°C)
Coefficient of Thermal Expansion (xflow)	ASTM E831	$57x10^{-06}$ µin/(in·°F)	$102x10^{-06}$ µm/(m·°C)

תקציר

מחקר זה עוסק ברובוט ה- RSTAR (Rising STAR), רובוט זעיר המסוגל לשנות את צורתו ולהזיז את מיקום מרכז המסה שלו. ה-RSTAR שייך למשפחת רובוטי ה-STAR המאופיינים ביכולת לשנות את זווית הפישוק של הרגליים ביחס לגוף, יכולת המאפשרת לו לנוע בתצורות שונות ולשנות את המישור בו נמצאות הרגליים. ה- RSTAR מצויד במנגנון נוסף, מנגנון FBEM, המאפשר לו לשנות את המרחק היחסי בין גופו לרגליו.

שילוב שני מנגנוני הקונפיגורציה הנ״ל מקנים ל- RSTAR יכולת להתגבר על מגוון מכשולים מאתגרים, לנוע על גבי תוואי שטח מגוונים ולטפס אנכית בתוך צינור או בין שני קירות. הרובוט יכול להאריך את גובהו ורוחבו פי שלושה ולהזיז את מרכז המסה שלו הן בכיוון אנכי לרצפה (מעלה –מטה) ובכיוון מקביל לרצפה (קדימה – אחורה).

ראשית יוצג המודל הקינמטי של הרובוט והניתוח הדינמי שבוצע כדי לשפר את תכן הרובוט ביחס לגרסאות קודמות ולהעריך את הדרישות מהמנועים. על סמך ניתוח זה, נבנה אב-טיפוס של הרובוט באמצעות הדפסה תלת-ממדית, על הרובוט בוצעו ניסויים מגוונים לבדיקת יכולות התנועה שלו במצבים שונים. ה-RSTAR הצליח לטפס מעל מכשולים שגבוהים יותר מקוטר הגלגלים שלו בעזרת שיטת טיפוס (Turtle Gait) שעושה שימוש ביכולות שינוי הצורה שלו. ניתן להתאים ל- RSTAR גלגלים רגילים, גלגלים משולשים או שילוב של השניים, וכך להעניק לו יכולת משופרת לנוע בסביבות קרקע שונות.

יחד עם זאת, במהלך הניסויים היה ניכר ששליטה בכל מנועי הרובוט במקביל היא משימה מאתגרת במיוחד כאשר יש צורך לבצע פעולות מורכבות יחסית כמו מעבר מכשולים. לכן נעשה שימוש בלמידת מכונה, בשיטות החיזוקים (Reinforcement Learning) על מנת ללמד את הרובוט כיצד להתמודד עם מעבר מכשולים בצורה אוטומטית, ללא התערבות מפעיל אנושי. נלמדו שלושה מכשולים טיפוסיים: תנועה במעבר צר, זחילה תחת פתח נמוך וטיפוס מעל מדרגה. במהלך דוח זה יוצג יישום של אלגוריתם למידה מסוג Q Learning בסביבת סימולציה ממוחשבת (UNITY™) עם מנוע פיזיקלי.

אלגוריתם הלמידה הצליח למצוא פתרונות עבור שלושת המקרים שנבחנו. לאחר קבלת תוצאות הלמידה, התוצאות הושוו לתוצאות שניתנו על ידי שישה מפעילים אנושיים בעלי רקע רלוונטי. התקבל כי התוצאות שהתקבלו על ידי האלגוריתם היו קצרות יותר מבחינת מספר הפעולות הנדרשות לביצוע, ביחס לתוצאות שניתנו על ידי המומחים האנושיים. בנוסף לכך עבור כל אחד מהמכשולים שנלמדו בוצע ניסוי בו נבחנו תוצאות הלמידה על רובוט ה-RSTAR האמיתי שנבנה. (ראה\י סרטונים מצורפים [24][48]).

אוניברסיטת בן גוריון בנגב

הפקולטה למדעי ההנדסה

המחלקה להנדסת מכונות

תכן, מידול ויישום של למידת מכונה בשיטת החיזוקים ברובוט משנה צורה המיועד למטרות חיפוש והצלה.

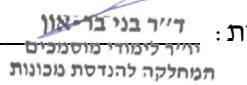
חיבור זה מהווה חלק מהדרישות לקבלת תואר מגיסטר בהנדסה

מאת: לירן יחזקאל

מנחה: ד"ר דוד זרוק

מחבר: לירן יחזקאל  תאריך: 31.05.2020

מנחה: דוד זרוק  תאריך: 31.05.2020

אישור יו"ר ועדת תואר שני מחלקתית: 

תאריך: 15.7.20

סיוון תש"פ

מאי 2020

אוניברסיטת בן גוריון בנגב

הפקולטה למדעי ההנדסה

המחלקה להנדסת מכונות

תכן, מידול ויישום של למידת מכונה בשיטת החיזוקים ברובוט משנה צורה המיועד למטרות חיפוש והצלה.

חיבור זה מהווה חלק מהדרישות לקבלת תואר מגיסטר בהנדסה

מאת: לירן יחזקאל

מנחה: ד״ר דוד זרוק

סיוון תש״פ

מאי 2020