# Evolving Adabot: A Mobile Robot with Adjustable Wheel Extensions

Anthony J. Clark
Department of Computer Science
Missouri State University
Springfield, Missouri, USA
AnthonyClark@MissouriState.edu

***Abstract*—Robots are increasingly being utilized in unstructured environments. Autonomous mobile robots are being assigned with tasks that are either too difficult or too dangerous for people. For instance, search and rescue robots can be deployed in unstable environments to aid in the search for disaster victims. In this paper, we propose a novel design for an autonomous mobile robot that can dynamically adjust traction during runtime. Our device, called Adabot meaning adaptive robot, is small, has a simple design, and can extend *wegs* from its wheels by adjustable amounts. We optimize both the morphology and the control parameters of Adabot using differential evolution, and our simulation results show that Adabot is effectively able to take advantage of both purely wheeled locomotion and legged-wheel locomotion by transitioning automatically between these two modes.**

## I. Introduction

Robots are often designed to operate in environments that are too hazardous or remote for humans. Examples include remote sensing (e.g., underwater exploration), assisting humans in dangerous occupations such as mining, and locating disaster victims (i.e., search and rescue). Each of these scenarios involve an environment that can be unpredictably dynamic with unknown structure. In this paper we focus on the search and rescue task. Specifically, we present a compact, autonomous mobile robot that has been designed to adapt to its current terrain in order to increase mobility in unforeseen environments. The device, which we call "Adabot" (for adaptive robot), is pictured in Figure 1. Adabot includes adjustable wheel extensions, called *wegs*, that can be quickly extended or retracted. Effectively, Adabot can operate on smooth wheels, extend the wegs a small amount to mimic tire studs, or extend the wegs completely so that the device operates similar to other mobile robots with legged-wheels [1].

Due to recent advances in sensing, computing, actuation, and battery systems, the past two decades have seen many advances in the field of mobile robots. In particular, mobile robots that combine wheeled and legged locomotion have received a great deal of interest [1–5]. The devices developed in these studies combine the simplicity provided by wheels with the ability of legged locomotion to navigate irregular, rough terrain with obstacles. Typically, such systems resemble a skid-steer mobile robot with four or six wheels, where in place of each wheel they have a rotating actuator that drives a rimless wheel with one or more spokes. Most devices, however, do not have the ability to operate purely as wheeled vehicles. Though, recently researchers have been developing robots that can dynamically switch between wheeled and legged-wheel modes [6–9].

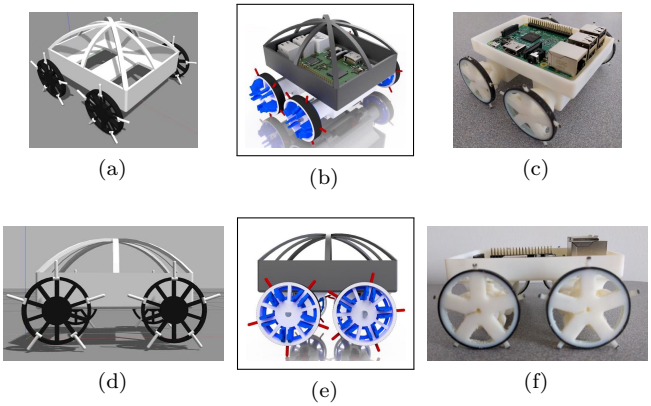



Figure 1. (a,d) Images of the Adabot being simulated in Gazebo, (b,e) CAD models for the Adabot design in Inventor, and (c,f) a 3D-printed prototype of Adabot. All images show Adabot with its wegs extended.

The mobile robot in Figure 1 more closely resembles these reconfigurable robots (i.e., it has the advantages of both pure wheeled and legged-wheel locomotion), however, with a significantly simpler mechanism for reconfiguration. With the wegs completely retracted, the robot behaves as any other skid-steer device [10] and has the advantages associated with wheeled locomotion: simple design, driving quickly and stably over flat terrain, simplified navigation, simple odometry calculations, relatively low vibrations and possibility of damage, etc. This mode does, however, come with several disadvantages: wheels can easily slip over loose or low friction terrain, wheels can also become stuck if they are wedged against an obstacle, and most importantly wheeled robots have trouble climbing over obstacles that are taller than the radius of the wheel [1]. The device presented in this paper can smoothly transition from using wheeled locomotion, to extending the wegs fully in rough

terrain, to retracting the wegs a small amount on somewhat uneven terrain, back to fully retracting the wegs when the surface is again level.

Although the extendable weg design has many benefits, it does require several design choices that drastically affect performance. Namely, the robot's morphology (i.e., shape, size, positioning, etc.) and control parameters must be specified. For this study we rely on an evolutionary algorithm to optimize these parameters. Like many evolutionary robotics (ER) studies, we perform evolutionary evaluations in simulation [11], and because we are utilizing the Robot Operating System (ROS) [12] on the physical device, we chose to leverage the expansive knowledge bases of ROS and Gazebo [13] for simulation. More specifically, we optimize the robot by simulating it in two distinct environments (a simple step world and a rocky terrain world). Evolving in simulated environments enables testing the device in harsh conditions, similar to what can be expected in real-life scenarios, without exposing it to unneeded risk. Our decision to use ROS/Gazebo comes with many advantages and technical challenges, which are discussed in Section IV.

The evolutionary results presented in this paper show the efficacy of the robot's design. Specifically, that the adjustable wegs enable the robot to effectively operate as if it were a wheeled or a legged-wheel robot. Additionally, the simple design, range of possible extension lengths, and small scale of our device provides an advantage over other similar robots. Another contribution of this work is the use of evolutionary optimization techniques in conjunction with ROS and Gazebo, which we believe will be of interest to the ER community. A Git repository[1] for the Adabot package can be found online. The remainder of this paper is structured as follows. In Section II we describe studies involving similar robotic systems. Sections III and IV describe the robotic platform and its simulation environment. Results of evolutionary experiments are discussed in Section V, and finally we present our conclusions in Section VI.

## II. Related Work

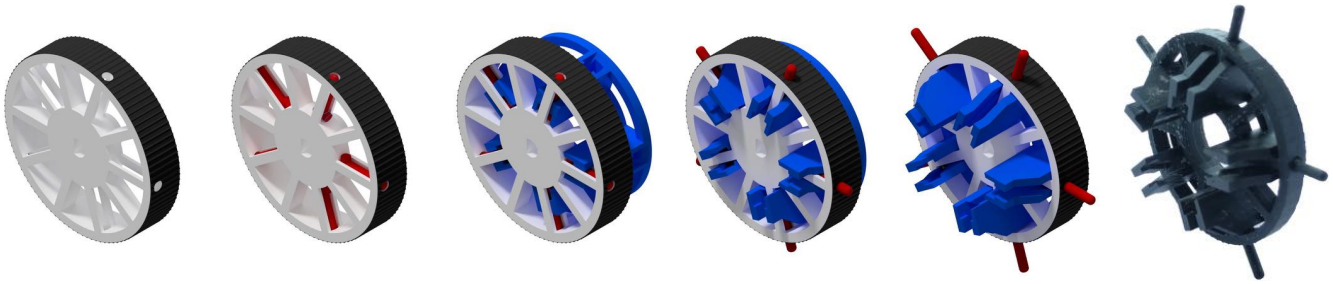
**Mobile Robots.** Robotics research focused on improving mobility has received a great deal of interest in the previous two decades. Researchers have discovered several unique methods for balancing the trade-offs between design simplicity and the increased complexity required to manage rough and unpredictable terrain. The simplest design is that of a standard wheeled robot. The inherent simplicity, however, comes at the cost of mobility. Purely wheeled robots suffer from poor mobility in uneven terrain. Thus, to improve performance, wheeled robots have been augmented with both active and passive suspensions. Some of the most effective, and complex, designs involve placing the wheels at the end of actively controlled legs. Take, for example, the Hylos [14] and NASA's Curiosity.

In an effort to retain the mobility of such systems while at the same time striving for simplicity (or avoiding unnecessary complexity), *legged-wheel* robots have been proposed. These devices, such as the Wheg 1 [1] and the ASGUARD [3], have rimless wheels in which the wheel spokes are what make contact with the ground. Compared with wheels, legged-wheels can drastically improved a robot's ability to navigate rough terrain and stairs. For example, when comparing a wheel and a legged-wheel with the same effective radius, the legged-wheel will be able to climb steps that are nearly the same height as the wheel diameter while the wheeled robot will likely encounter difficulty when the step height is above the wheel's radius [1, 7]. A subset of legged-wheel robots incorporate only a single spoke (leg) on each wheel that rotates around a central axis. The RHex [2] and VelociRoACH [4] are examples of hexapod-like mobile robots that use a single spoke per wheel to mimic animal-like gaits. Both of these devices derive their motion from the insect-like tripod gait, which has proven to be very effective in rough terrains. Similar to legged-wheels are *wheeled-legs*, which include a powered wheel at the end of each spoke. These devices aim to incorporate the benefits of purely wheeled and legged-wheel locomotion, however, this comes at the cost of increased complexity and larger wheels [7, 15].

More recently, the research community has put an emphasis on transformable legged-wheels. These devices include a mechanism for reconfiguring the wheel (i.e., significantly changing its shape) from a purely wheeled mode to a legged-wheel mode [6, 9, 16, 17]. The device presented in this study is most similar to these transformable robots. However, compared with other devices, the wheel presented in this study is able to exhibit a range of different behaviors, rather than purely wheeled or fully legged-wheeled. Additionally, our robot has a simpler transforming mechanism that requires only a single actuator to extend the wegs of all four wheels–though, the version presented here includes a linear servo for each wheel to maximize the flexibility of our design. The Adabot also has the advantage of few moving parts and joints, and the ability to more easily miniaturize the design for applications in which the robot needs to be smaller. For example, it is beneficial to have a small, lightweight robot to lower the possibility of causing further damage to the environment or injury to a person. However, our weg extensions have a relatively low maximum extension distance, roughly equal to the difference between the radius of the wheel and the radius of the axle (see Figure 5).

**Evolutionary Robotics.** Roboticists have often drawn inspiration from nature to design more robust systems. Likewise, the field of evolutionary computation (EC) applies concepts from natural evolution to optimization algorithms. Evolutionary robotics is a branch of both

[1]Git repository: https://github.com/anthony-jclark/adabot



(a) (b) (c) (d) (e) (f)

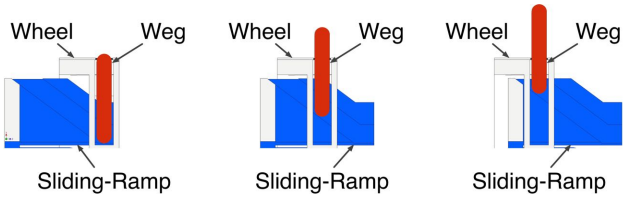
Figure 2. CAD models showing the (a) wheel and its empty weg channels, (b) wegs in the fully retracted position, (c) retracted wegs and the sliding-ramp, (d) wegs and sliding-ramp in a partially extended position, and (e) wegs and sliding-ramp in a fully extended position. An image of the 3D printed prototype is shown in (f).

robotics and evolutionary computation in which evolutionary algorithms are applied to optimize both the morphology and control of physical systems [11]. For example, Iwasa *et al.* [18] used a genetic algorithm and knowledge transfer to improve the mobility of a hexapod walking in rough terrain. Similarly, the robot presented in this study is optimized using an evolutionary algorithm called differential evolution (DE) [19]. In prior work [20], we investigated the use of DE to improve the adaptability of a robotic fish by optimizing both the physical properties of the robot's tail fin as well as its control parameters.

## III. Robot Platform

**Adabot Hardware.** The Adabot is a compact, autonomous mobile robot. The prototype pictured in Figure 1 is controlled by a Raspberry Pi 3 Model B (RPi), which was chosen due to its low cost and ability to effectively run ROS. In addition to the RPi, the Adabot includes an A-Star 32U4 micro-controller board to offload some of the sensor processing. Each wheel is directly driven by a DC gear-motor with magnetic wheel encoders, and each set of wheel wegs is extended/retracted by a 2.3 gram linear servo. For sensors, the Adabot includes three forward facing IR sensors, a 9-axis IMU (gyro, accelerometer, and magnetometer), as well as a 2.4 GHz wireless communication module. Finally, the Adabot is powered by a 2200 mAh NiMH battery pack and includes the necessary voltage regulators and motor controllers to interact with the sensors and actuators. The prototype device has been fabricated with the aid of 3D printing technology, though parts on the finalized device are being cast from plastic resins. The 3D printing process enables quick validation of evolved solutions.

Figure 2 shows how the wheels have been designed. In the figure, the wheel's radius and depth are 2 cm and 0.4 cm, respectively, and the wegs can extend from the wheel rim approximately 1 cm. The mechanism comprises four main components: a wheel (white with black tread), several wegs (red), a sliding-ramp (blue), and a sliding-push (not shown in this figure, see Figure 4). Essentially, as the sliding-ramp is pushed outward the wegs slide




(a) (b) (c)

Figure 3. A cross-section view of a single wheel channel showing the wegs in (a) the retracted state, (b) a partial extended state, and (c) the fully extended state. As the sliding-ramp (blue) moves from left-to-right the weg (red) is pushed up the wheel channel.

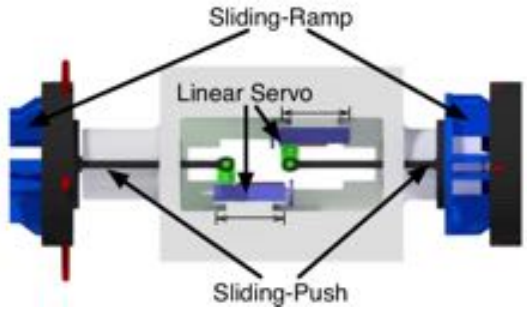



Figure 4. A top-down view of an Adabot axle cavity (with the RPi removed). Servomotors connected to the sliding-push components are responsible for pushing and pulling the sliding-ramp, which in turn extends and retracts the wegs.

radially out from the wheel's axle. Figure 3 depicts a cross-sectional drawing of the sliding-ramp, wheel, and wegs. This mechanism is simple, effective, and can be scaled up or down in size.

Each sliding-ramp is pushed and pulled outward and inward by a sliding-push component, as shown in Figure 4. The sliding-push components are attached to linear servos situated in the space between the wheel motors. This design is compact and also has considerable flexibility since each wheel and each set of wegs can be controlled independently.

To maximize extension of wegs while also allowing for a minimal wheelbase (i.e., the distance between front and

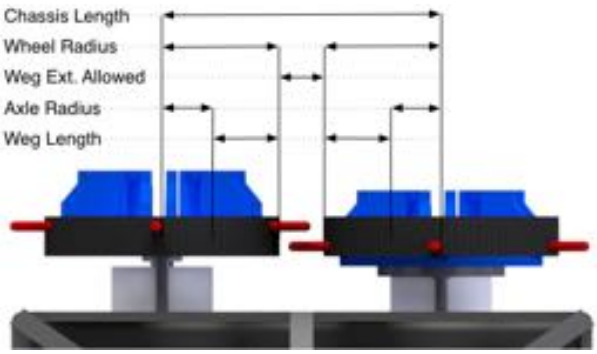



Figure 5. A depiction of the weg offset in the wheel's outer rim. In the wheel on the left (the front wheel), the wegs protrude closer to the outer edge of the wheel, whereas the opposite is true for the wheel on the right (the rear wheel). This enables wegs to extend a greater distance without colliding while at the same time reducing the maximum required chassis length (i.e., the wheelbase). Also depicted in the figure are the two evolved parameters (chassis length and wheel radius) that constrain the maximum allowed wheel extension distance, which can be less than the length of a weg.

rear axles), wegs do not extend from the center of a wheel's outer rim. As shown in Figure 5, the front wheel wegs extend from nearer the outer edge and the rear wheel wegs extend from nearer the inner edge of the wheel rim. Minimizing the wheelbase is important for skid steer drives as the relationship between wheelbase and wheel-track (i.e., the distance between tires on a single axle vector) impacts the amount of vibration/skidding during a zero-degree turn. Specifically, as the wheelbase is increased relative to the wheel-track so do the frictional forces that cause vibration.

Although the prototype has a wheel radius of 2 cm, it should be noted that as the wheel radius is increased so to is the maximum possible weg extension distance. This point is discussed in more detail in the next section where we discuss the evolved parameters.

**Adabot and ROS.** RPis are a powerful option for a small embedded system. The Raspberry Pi 3 Model B has a 1.2GHz, 64-bit CPU and 1 GB RAM, as well as built in Bluetooth and a 40-pin GPIO header among other features. The RPi enables use of ROS, which is a set of libraries and middleware that aid the in the development of robotic systems. At its core, a ROS controlled system launches several software nodes that communicate with one another via a publisher/subscriber message passing (or through services and actions). Each node has a single well-defined task, and nodes are designed to be as reusable as possible. For example, one node might read IMU data from a specific IMU sensor chip over an $I^2C$ interface, and then publish this data for use by other nodes.

In addition to nodes that read from sensors and send information to motor controllers, we are using several custom, single purpose nodes and two commonly used nodes from the ROS community. First, we have a node that generates odometry from our wheel encoder data. This data and the IMU data are then fused by the *robot_localization* [21] package, which provides localization information (positions, velocities, and accelerations) using a Kalman filter. We then have a custom ROS node that reads the current velocity from the localization data and smooths it using a windowing technique so that it can be used by the higher level controller (described in the next section). Finally, the *ros_control* package is utilized to manage PID controllers for each of the four DC motors connected to the wheels.

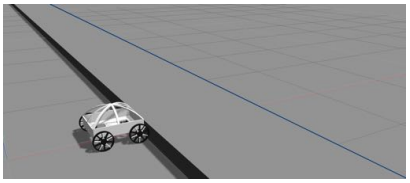
## IV. Simulation and Evolution

**Simulation Environment.** Gazebo is an open source simulation package with up-to-date ROS integration; both ROS and Gazebo are managed by the Open Source Robotics Foundation (OSRF). Although Gazebo can act as a stand-alone simulator, when used in tandem with ROS it behaves like any other ROS node. Specifically, a Gazebo simulation instance acts as a drop-in replacement for a real robot and its physical environment. Essentially, as a simulated vehicle operates in its environment Gazebo will publish raw sensor data (in an identical format to what is published by a node reading from a real sensor) and with the *gazebo_ros_control* package, *ros_control* commands will be acted upon by simulated actuators. Another key benefit of Gazebo is that its sensor plug-ins (like the IMU sensor used in this study) are thoroughly tested and have built-in noise models so that they more closely mimic real devices. One additional responsibility of the Gazebo node is to publish the current simulation time, which all nodes use in place of real time.

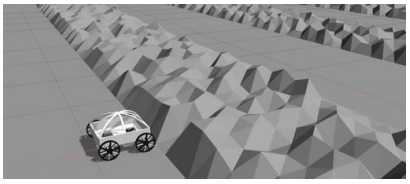
To test and evolve Adabot, we have created two simple environments: a step world, and a rocky world. The step world, depicted in Figure 6 (a), serves two purposes: (1) it is a minimal environment we can use to investigate the Adabot weg mechanism, and (2) it provides a baseline comparison for experiments that occur in more complex environments. In particular, we can compare optimized parameters from the step world to the rocky world to investigate whether a single set of morphological parameters is likely best for all environments or if different types of obstacles and terrain require different parameter values.

Figure 6 (b) shows the rocky world environment. This environment comprises three randomly generated "rocky" regions separated by flat plains, where peaks in the rocky regions are evenly distributed in a 1 by 1 cm grid pattern and the height of each peak is a randomly generated. This world is randomly generated prior to simulation and then kept constant throughout all replicate experiments. The rocky world includes a more difficult terrain that better reveals the relative strengths and weaknesses of the Adabot design.

**Evolutionary Optimization.** We chose the differential evolution (DE) algorithm for this study, and we are using the Distributed Evolutionary Algorithms in Python (DEAP) software package [22]. DE is similar to a conven-



(a) Step World



(b) Rocky World

Figure 6. The step world (a) contains a single obstacle that is 3.5 cm tall and 25 cm wide. The rocky world (b) contains three rocky regions that are 25 cm wide with 25 cm gaps between them, where each region comprises random-height peaks of at most 6 (near), 8 (middle), and 10 cm (far).

tional steady-state genetic algorithms, however, its genetic operators have proven to be particularly effective for real-valued problems [19]. For this study, we have implemented dither for the mutation factor [23]. The objective for DE is to maximize average velocity of the Adabot in the x-axis direction (i.e., straight ahead from its initial orientation). During any given simulation, the Adabot is being controlled by the state machine depicted in Figure 7. Along with several morphological parameters, the values that dictate transitions in this state machine are optimized by DE. After testing several variants, we have configured DE as *DE/rand/1/bin* (see Storn *et al.* [19] for a description of the nomenclature).

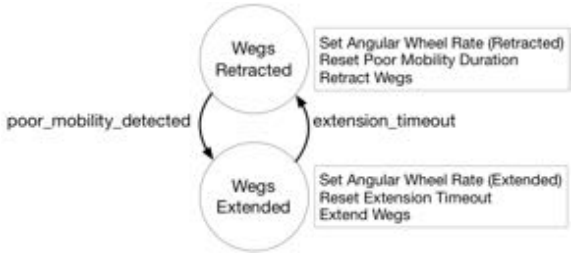



Figure 7. The depicted state machine automatically extends and retracts the wegs based on sensor feedback. The robot starts in the retracted state and then transitions to the extended state once the poor mobility flag has been set, which is based on several evolved parameters. The transition back to the retracted state is based on an evolved timeout period.

A full listing of the evolved parameters can be found in Table I. We evolve four morphological parameters: the chassis length and width, the radius of the wheels, and the number of wegs per wheel. For control, we evolve five parameters. First, an angular wheel rate (i.e., a rate independent of the wheel radius, meaning on a flat surface a larger wheel will produce a higher linear velocity) for the fully retracted weg-state, and an angular wheel rate for when the wegs are extended. These wheel rates serve two purposes: first, it may be beneficial to drive at a different speed with the wegs extended or retracted (for example, it may be more stable to drive slower with the wegs extended), and second, the top speed of the robot with the wegs extended is lower to reduce the risk of damage to the wegs. We also evolve a poor mobility threshold factor and a poor mobility duration threshold, which are used to detect how long the Adabot has not been achieving its directed speed and then trigger extension of the wegs. For example, when Adabot reaches an obstacle with its wegs retracted, the *robot_localization* package will indicate that it is no longer driving at the directed speed (the current speed will be different from the desired speed by at least the *Poor Mobility Threshold Factor*). Once the Adabot detects that such a condition has transpired for a specified period of time (*Poor Mobility Duration Threshold*) its wegs are extended. Specifically, they are extended by some percentage of their maximum possible amount (*Weg Extension Percentage*) for a specified amount of time (*Weg Extension Duration*), which are the final two evolved parameters.

Table I
Evolved Adabot Parameters

| **Description** | **Range** |
|---|---|
| Chassis Length | 6 to 15 cm |
| Chassis Width | 6 to 15 cm |
| Wheel Radius | 1 to 3 cm |
| Wegs Per Wheel | 0 to 7 |
| Angular Wheel Rate (Retracted) | 0 to 9 rad s$^{-1}$ |
| Angular Wheel Rate (Extended) | 0 to 4 rad s$^{-1}$ |
| Poor Mobility Threshold Factor | 0 to 1 |
| Poor Mobility Duration Threshold | 0 to 7 s |
| Weg Extension Percentage | 0 to 100 % |
| Weg Extension Duration | 0 to 30 s |

The maximum possible weg length, and therefore the maximum weg extension amount, is based on the wheel radius (a configurable value) and the axle radius (a fixed value). Additionally, the amount of space between wheels (the amount of weg extension allowed) depends on the wheel radius and the chassis length (i.e., the wheelbase). Figure 5 depicts these parameters. If the evolved parameters result in wegs that collide with another wheel, then we add a constraint violation to the evolved individual. Specifically, we use the following equations:

$$weg_{length} = wheel_{radius} - axle_{radius}$$
$$extension_{allowed} = chassis_{length} - wheel_{radius} * 2$$
$$extension_{evolved} = weg_{length} * extension_{percent}/100.0$$
$$violation = extension_{evolved} - extension_{allowed}$$

where $weg_{length}$, $extension_{allowed}$, and $extension_{evolved}$ denote the full weg extension possible for a single wheel, the amount of space available between wheels, and the evolved extension amount, respectively. If *violation* is greater than zero then the evolved individual has a constraint violation and the weg extension percentage is reduced to the maximum allowed value. Once the weg extension percentage is reduce, fitness is evaluated as normal. Finally, when comparing individuals based on fitness and constraint values, if they are both constraint-free then the higher fitness value is preferred, if one individual has a constraint violation then the non-constraint individual is preferred, if they both have a constraint violation then the lower violation individual is preferred. This is a simple and effective method for handling constrained values among evolved parameters [24].

**Evaluation.** To evaluate an individual, we configure a Gazebo simulation with the individual's parameter values and repeat the same simulation five times. It is important to repeat the simulation multiple times as an inherent property of ROS/Gazebo simulations is that they are not deterministic (due to the message passing system and lack of reset functionality for all plug-ins and external nodes). An additional benefit of repeated evaluation is that it reduces the chances of a selection bias attributed to evolved solution being "lucky" during fitness evaluation [25]. We settled on five repeated simulations after some experimentation and finding the best balance between execution time and repeatability. The final fitness value is calculated as the average velocity in the x-direction once outliers are discarded using the median filtering method. One positive side-effect of evaluating each genome multiple times is that the fitness function applies an evolutionary pressure on producing genomes that are repeatably high fitness. Effectively, the fitness function prefers solutions that are not easily knocked off a straight-ahead course by obstacles in the environment.

## V. Results

In this section we describe and compare results from two experiments: evolving the Adabot in a rocky world and in a step world. For both experiments, we configure DE parameters after experimenting with the values recommended by Storn *et al.* [19]; for both experiments, we use a population size of 40, a dithered mutation factor from 0.5 to 1.0 (a random value is generated each generation), and a crossover rate of 0.9. Additionally, each experiment comprises 40 replicates (initialized with different random number generator seeds) evolving for at most 50 generations. We include an early stopping criteria to prevent wasted simulation time–if the maximum fitness does not improve for 10 consecutive generations and the current generation is greater than 25, then the replicate is stopped. On average, an experiment is stopped-early at generation 34 and 43 for the rocky and step worlds, respectively. The duration of trials in the rocky and step environments are 30 seconds and 20 seconds, respectively, and fitness is calculated as described in the previous section. Evolutionary trajectories for these experiments are shown in Figure 8, and videos of optimized behaviors for the rocky[2] and step[3] environments can be found online.

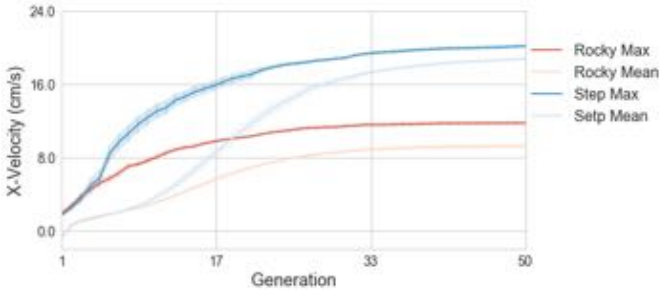


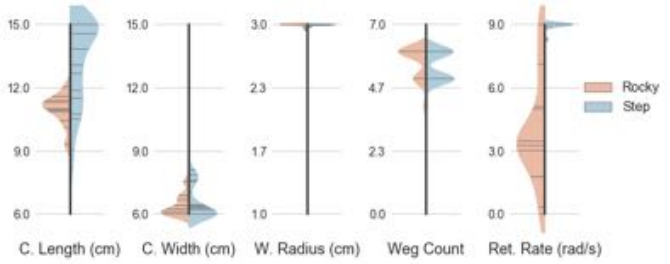

Figure 8. The figure above shows the max and mean fitness trends (measured as velocity in the x-axis) for both replicated experiments as they improve over the generations. The shaded regions and vertical bars near each line indicate confidence intervals of one standard deviation from the mean. A maximum fitness of value of 11 and 20 $\mathrm{cm\,s^{-1}}$ is achieved on average in the rocky and step environments, respectively.

As demonstrated by the figure, both replicated experiments converge on a final fitness value, and it can be seen, that on average, fitness values attained in the step environment are higher. Furthermore, the final mean fitness values for the step world (19 $\mathrm{cm\,s^{-1}}$) are significantly higher when compared to those of the rocky world (9.3 $\mathrm{cm\,s^{-1}}$). These characteristics indicate that the rocky world is a more difficult challenge for the simulated Adabot.

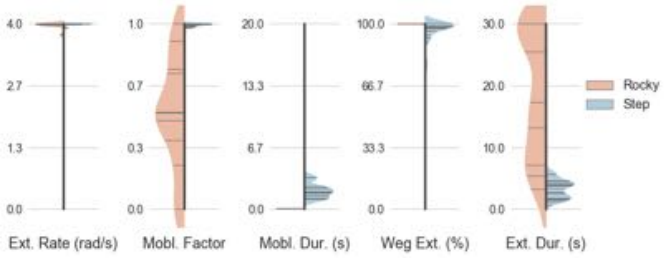
Considering the evolved parameters, the highest attainable velocity is 24 $\mathrm{cm\,s^{-1}}$. This speed assumes that the Adabot is driving over flat terrain with the largest wheel size and the fastest allowed angular wheel rate for the retracted wegs state. It is also worth noting that the fastest possible velocity with the wegs extended is approximately 20 $\mathrm{cm\,s^{-1}}$, which is based on an effective wheel radius that includes the extended wegs (i.e., wegs extended 2.2 cm in addition to a 3 cm wheel radius). Thus, the evolved velocities are roughly 80 and 39 % of the maximum value, respectively. We consider this to be a good result because both environments require the Adabot to extend its wegs to move forward in the x-direction, and the trigger to extend the wegs is based on the robot becoming "stuck" in a low mobility state. Hence, it is not possible to exactly reach the maximum 24 $\mathrm{cm\,s^{-1}}$.

To identify behaviors of the robot we look at distributions for the evolved parameters. As shown in Figure 9 (a) the morphological parameters (the first four plots in (a)) converge to final values for each environment. The evolved chassis lengths (11 cm) and widths (6 cm), and the wheel radius (3 cm) are roughly the same for both environments.

[2]Rocky environment: https://youtu.be/A5NfjCTC7vQ
[3]Step environment: https://youtu.be/pMf8tchKE6g




(a)




(b)

Figure 9. Distributions for the evolve parameters listed in Table I. Each plot shows the distribution of the top 100 individuals (across all replicates) from each environment. The left-half (appearing in red) shows values from the rocky experiments and the right-half (shown in blue) for the step environment. Horizontal lines denote an exact evolved parameter value, the y-axis limits are the limits allowed during evolution.

Though, longer chassis are found to be effective in the step world. The chassis values are lower in the rocky world since larger chassis would increase the chances of becoming high centered on uneven terrain, which is less of a concern in the step world. As for the number of wegs per wheel, it appears that for both environments roughly 5 or 6 wegs per wheel is optimal. This is intuitive, as having an increased number of wegs improves he ability of Adabot to climb obstacles, but too many will effectively results in just a larger wheel that does not provide an advantage for climbing obstacles.

For control, the Adabot evolved similar strategies for the two different environments. Specifically, high and medium values for the poor mobility threshold factor (the second distribution plot in Figure 9 (b)) and low values for poor mobility duration threshold (the third distribution) indicate that the robot will quickly extend its wegs in reaction to an obstacle. Essentially, the controller will deem any short-in-duration variation in speed to indicate poor mobility. Furthermore, once the wegs are extended the Adabot moves at its maximum allowed speed of 4 $\mathrm{rad\,s^{-1}}$, which corresponds to roughly 20 $\mathrm{cm\,s^{-1}}$.

Weg extension duration converges in the step environment (they are retracted as soon as Adabot climbs the step), but not in the rocky environment (the final distribution in Figure 9 (b)). This is likely because in the rocky world Adabot quickly returns to the wegs extended state, which reduces the need for evolving long extension durations. For the same reason, there does not appear to be a high pressure on evolving a high speed in the retracted wegs state for the rocky world, as the Adabot is rarely in this state. Finally, the fourth distribution in Figure 9 (b) shows that the wegs are fully extended in both environments, and we should note that the chassis lengths evolved for both environments are long enough to accommodate full extension of the wegs (as would any chassis length over 8.2 cm). Meaning that the most optimal solutions in terms of fitness do not contain any constraint violations.

## VI. Conclusion

We have presented Adabot, an adaptive, autonomous mobile robot. Adabot's main advantage is a simple mechanism for transforming its smooth wheels into legged-wheels. Compared with other devices with similar design goals, Adabot can exhibit a wider range of different locomotion modes. Specifically, Adabot can operate with purely wheeled locomotion, wheels that appear to have studs, or fully legged-wheels. We evolved Adabot's morphology and control parameters so that it could achieve a maximum velocity when faced with rough terrain. Our results show that Adabot can effectively detect poor mobility using localization data and then respond by transitioning from a retracted weg state to an extended weg state. Our results also show that some morphological characteristics and control strategies must be tuned to the type of environment in which the Adabot is operating. Our ongoing work includes prototyping compliant wegs, evolving Adabot against more difficult tasks (such as way-point following), and generating controllers based on fuzzy-systems that extend the wegs by different amounts dependent on current conditions.

## Acknowledgment

The authors gratefully acknowledge the contributions and feedback on the work provided by Megan Clark and Professors Philip McKinley and Jared Moore. Additionally, we would like to thank Keith Cissell, Dillon Flohr, Dersham Schmidt, and Daniel Warken from Missouri State University.

## References

[1] R. D. Quinn, J. T. Offi, D. A. Kingsley, and R. E. Ritzmann, "Improved mobility through abstracted biological principles," in *Proceedings of the 2002 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, vol. 3, 2002, pp. 2652–2657.

[2] U. Saranli, M. Buehler, and D. E. Koditschek, "RHex: A simple and highly mobile hexapod robot," *The International Journal of Robotics Research*, vol. 20, no. 7, pp. 616–631, 2001.

[3] M. Eich, F. Grimminger, and F. Kirchner, "A versatile stair-climbing robot for search and rescue applications," in *Proceedings of the 2008 IEEE International Workshop on Safety, Security and Rescue Robotics*, Oct. 2008, pp. 35–40.

[4] D. W. Haldane, K. C. Peterson, F. L. G. Bermudez, and R. S. Fearing, "Animal-inspired design and aerodynamic stabilization of a hexapedal millirobot," in *Proceedings of the 2013 IEEE International Conference on Robotics and Automation (ICRA)*, May 2013, pp. 3279–3286.

[5] G. Kenneally, A. De, and D. E. Koditschek, "Design principles for a family of direct-drive legged robots," *IEEE Robotics and Automation Letters*, vol. 1, no. 2, pp. 900–907, Jul. 2016.

[6] 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 Transactions on Robotics*, vol. 30, no. 6, pp. 1487–1498, Dec. 2014.

[7] C. Zheng, J. Liu, T. E. Grift, Z. Zhang, T. Sheng, J. Zhou, Y. Ma, and M. Yin, "Design and analysis of a wheel-legged hybrid locomotion mechanism," *Advances in Mechanical Engineering*, vol. 7, no. 11, 2015.

[8] B. P. Rhoads and H.-J. Su, "The design and fabrication of a deformable origami wheel," in *Proceedings of the ASME 2016 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference (IDETC/CIE)*, Aug. 2016.

[9] W. H. Chen, H. S. Lin, Y. M. Lin, and P. C. Lin, "Turboquad: A novel leg-wheel transformable robot with smooth and fast behavioral transitions," *IEEE Transactions on Robotics*, vol. PP, no. 99, pp. 1–16, 2017.

[10] A. Mandow, J. L. Martinez, J. Morales, J. L. Blanco, A. Garcia-Cerezo, and J. Gonzalez, "Experimental kinematics for wheeled skid-steer mobile robots," in *Proceedings of the 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct. 2007, pp. 1222–1227.

[11] F. Silva, M. Duarte, L. Correia, S. M. Oliveira, and A. L. Christensen, "Open issues in evolutionary robotics," *Evolutionary Computation*, vol. 24, no. 2, pp. 205–236, Jun. 2016.

[12] M. Quigley, K. Conley, B. P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "ROS: An open-source robot operating system - Willow Garage," in *Proceedings of the 2009 IEEE International Conference on Robotics and Automation (ICRA) Workshop on Open Source Robotics (OSS)*, May 2009.

[13] N. Koenig and A. Howard, "Design and use paradigms for Gazebo, an open-source multi-robot simulator," in *Proceedings of the 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, vol. 3, Sep. 2004, pp. 2149–2154.

[14] C. Grand, F. Benamar, F. Plumet, and P. Bidaud, "Stability and traction optimization of a reconfigurable wheel-legged robot," *The International Journal of Robotics Research*, vol. 23, no. 10-11, pp. 1041–1058, 2004.

[15] L. M. Smith, R. D. Quinn, K. A. Johnson, and W. R. Tuck, "The Tri-Wheel: A novel wheel-leg mobility concept," in *Proceedings of the 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sep. 2015, pp. 4146–4152.

[16] Y. She, C. J. Hurd, and H. J. Su, "A transformable wheel robot with a passive leg," in *Proceedings of the 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sep. 2015, pp. 4165–4170.

[17] Z. Wei, G. Song, Y. Zhang, H. Sun, and G. Qiao, "Transleg: A wire-driven leg-wheel robot with a compliant spine," in *Proceedings of the 2016 IEEE International Conference on Information and Automation (ICIA)*, Aug. 2016, pp. 7–12.

[18] M. Iwasa, T. Obo, and N. Kubota, "Motion generation of multi-legged robot by using knowledge transfer in rough terrain," in *Proceedings of the 2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, Dec. 2016, pp. 1–5.

[19] R. Storn and K. Price, "Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.

[20] A. J. Clark, P. K. McKinley, and X. Tan, "Enhancing a model-free adaptive controller through evolutionary computation," in *Proceedings of the 2015 ACM Genetic and Evolutionary Computation Conference (GECCO)*, Madrid, Spain, Jul. 2015, pp. 137–144.

[21] T. Moore and D. Stouch, "A generalized extended Kalman filter implementation for the Robot Operating System," in *Proceedings of the 13$^{th}$ International Conference on Intelligent Autonomous Systems (IAS-13)*, Springer, Jul. 2014.

[22] F.-A. Fortin, F.-M. De Rainville, M.-A. Gardner, M. Parizeau, and C. Gagne, "DEAP: Evolutionary algorithms made easy," *Journal of Machine Learning Research*, vol. 13, pp. 2171–2175, Jul. 2012.

[23] S. Das and P. N. Suganthan, "Differential evolution: A survey of the state-of-the-art," *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 1, pp. 4–31, Feb. 2011.

[24] E. Mezura-Montes, C. A. Coello Coello, and E. I. Tun-Morales, "Simple feasibility rules and differential evolution for constrained optimization," in *Proceedings of the 2004 Mexican International Conference on Artificial Intelligence (MICAI)*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 707–716.

[25] E. L. Ruud, E. Samuelsen, and K. Glette, "Memetic robot control evolution and adaption to reality," in *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, Dec. 2016, pp. 1–7.