ORIGAMI WHEEL DESIGN FOR ADVANTAGEOUS TRAVERSAL OF UNKNOWN TERRAIN

Luc Bettaieb
Department of Electrical Engineering and Computer Science
Case Western Reserve University
Cleveland, OH, USA

Xueqing Hu and Xiangyi Cheng
Department of Mechanical and Aerospace Engineering
Case Western Reserve University
Cleveland, OH, USA

***Abstract*—A deformable origami wheels robot is designed and constructed in this paper. The wheels are folded using a ninja star pattern which can make the wheels achieve two shapes, an octagon-shape and a spiky-shape. Octagon-shape wheels are suitable for moving on the flat and smooth ground while spiky-shape wheels are for rough ground. The flexibility of the wheels increases the possibility of robot traverses advantageous unknown terrain.**

Keywords—origami; deformation wheels; ninja-star; robot; ros

I. INTRODUCTION

Origami is an artform centered around folding paper originating from Japan. The materials chosen for folding can be any flat foldable materials which can hold creases. With the advent of a new era of commercial space exploration, there is an emerging market of low cost, small launch systems for launching payloads into low Earth orbit and beyond for research purposes. Furthermore, small origami robots have potential to be an advantageous platform for exploration of new environments because of some origami structures transformability, flexibility, and rigidity. Furthermore, any useful payload on a rocket, if possible, needs to be as small as possible to save space. Therefore, origami robots have been attracting more attention lately.

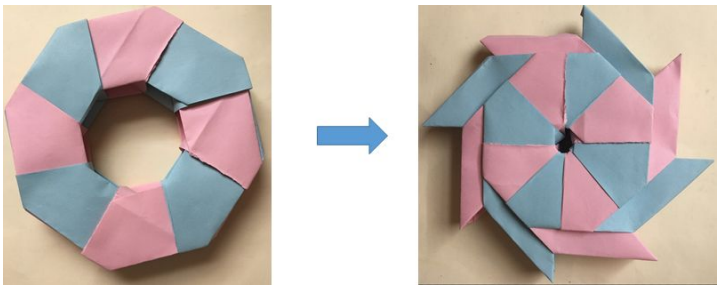
In the field of origami wheel design, Lee et al. have developed a series of origami wheels. For example, his team developed a deformable wheel robot based on the "magic-ball" pattern [1] which was developed in 2013 to get through small spaces by shrinking the wheels. There are three states of the folded wheels that may be actuated using strings. The creases on the paper formed various small triangles to generate a sphere-like structure, and spring was put into the wheel to control the deformation of the wheel's size.

In another paper [2] by Lee et al., the magic-ball pattern was used again. However, rather than using a spring inside the wheel, a sliding shaft and cable which are controlled by a motor are used. When the motor pulls the wire installed inside the wheel, the wheel diameter becomes larger. The wheel returns to the original shape when the motor unwinds the wire because of the rubber band around the wheel. The advantages of this design are that the wheels have more actuation states and the structures controlling the deformation are easier to implement. However, the magic-ball pattern is difficult to fold and the folding paper may break easily because of too many creases on one paper [7]. Thus, some other supporting tools need to be used when folding or the rate of failure is high.

This team also came up with a different pattern deformable-wheel robot [3] by using soft materials, rigid films and flexible films, which allowed the robot to be flexible. The cylindrical shape of the wheels comprised of composite segments could achieve three types of movement: (1) Original movement in cylinder-shape at high speed on flat ground; (2) Caterpillar-like movement through a narrow gap; (3) Legged-wheel motion for climbing a stair.

Another type of origami wheel, developed by a team led by Samuel M. Felton [5], responds to an increase in torque by reducing its radius though the spring-like properties.

Based on the above research results and our own origami folding experience, we chose the idea of a ninja star [6] as the prototype of our origami wheel which is comprised with eight same parts. The three dimensional ninja star can transfer between an octagon-shape wheel and a spiky wheel which, if used as a wheel, could enable a robot to gain more traction (Fig. 1). Considering the mechanical structure, the wheel's shape can be controlled by linkages and gears actuated by servos. Also, motors are applied to control the rotation movement of the wheels. All the servos and motors are independent so that the deformation and rotation can't be influenced by each other. We think a deformable wheel under this pattern can be suitable for traversing different types of terrain more easily such as on smooth and flat ground, sandy and rocky ground or more mutable terrain such as loose dirt, and light gravel.



(a) Octagon-shape Wheel (b) Spiky wheel

Fig. 1. Two status of three dimensional Ninja Star

II. DESIGN

A. Mechanical Design

1. Wheel Design

A three dimensional ninja star structure is used as the basis of the wheel. The plastic material takes advantage of better stability and load affording ability when compared to paper. So transparency film designed for laser printers was used for the wheels. The ninja star is assembled by eight identical parts. The shape of each part is a three dimensional parallelogram. The width of each parallelogram is two inches. To assemble two parts together, two grooves are cut near the edges which are in the opposite sides and attached to another part in one parallelogram. The grooves make sure two parts stay together while allowing for one degree of freedom to exist between each parallelogram structure. Following this rules, we connected all eight parallelograms together so every part has two grooves in which reside the sheets of another part. The wheel folded by plastic material is shown in Fig. 2.

(a) Octagon-shape Wheel (b) Spiky wheel

Fig. 2. Wheel structure

2. Actuator design

To save the for wheels deformation, a thin axis is chosen to go through the wheel. For one wheel, a motor controls the rotation of the axis to drive the robot forwards. On each side of the wheel, gear 1 is fixed on the axis whose radius is 1.18 inch (Fig. 2a) which means the gear rotates with the axis. Gear 2 is attached with the servo so that the servo actuates the rotation motion of the gear independently. The radius of gear 2 is 0.787 inch as shown in Fig. 3. The servo is installed on a laser-cut wood plate. The wood plate can rotate by the axis even if the axis is stationary. There are five holes on the wood plate. The biggest one is in the middle is for axial rotation. The others are in the same size which are for four linkages, as shown in Fig. 4. The linkages connect the wood plate and the plastic wheel.

Fig. 3. Gears

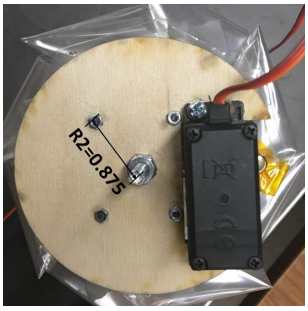


Fig. 4. wood plate and servo

In this way, forward moving motion and wheels deformation won't conflict each other. The initial shape of the wheels is an octagon. When the wheels go forward, the gears are locked so that the size of the wheels is not changed. Once the command of servo rotation is sent, the gears start to rotate and gear 1 causes the wooden plate to rotate. The linkages attached on the wood plates pull four parts of the wheels to slide along grooves in the others' edges with the wood plates rotation. Therefore, the wheel turns into its spiky state. To transform the wheels back into their original state, the opposite rotation command is sent to the servos. The linkages will return to the initial state by the wood plate's gear-actuated rotation.

B. Electronics and Control

The control system of our robot was comprised of several different pieces of hardware (Fig. 5). Firstly, there was the Raspberry Pi which ran Ubuntu ARM 14.04 with the ROS Indigo Igloo robotics middleware platform. On the Raspberry Pi, software ran to interface the ROS control structure with the hardware that made the robot move and sense. To provide this interface from software to hardware and vice versa, the WiringPI library was used.From the Raspberry Pi's general purpose I/O (GPIO), pulse width modulated signals were sent to a custom made LM293B control circuit for variable motor speed. Additionally, a signal could be sent to the Pololu Micro Maestro (Fig. 6) to toggle the state of the servos. Finally, encoders attached to the wheel motors were hooked up to the GPIO of the Raspberry Pi. The LM293B control circuit was first constructed on a breadboard for testing. After successfully being table to control motors going forwards and backwards, pulse width modulated (PWM) control signals were introduced into the input. They were successfully able to control the speed of the motor from a slow pace to max speed.

The control circuit (Fig. 8) was designed according to schematics provided by the datasheet for the IC, however, instead of one motor being used, two were controlled from one chip by using the other eight pins on the right hand side of the package. Additionally, pull-down resistors were used on the inputs in order to make sure that the input did not float and cause problems with controlling the motors. However, because of a currently unknown

problem, the motors were only able to be controlled to go forward. More debugging will need to be done for this to be resolved.

Lastly, there is the Pololu Micro Maestro. Initially, we hooked servos up to the board and attempted to control them manually using the built-in software. We found, however, that problems quickly arose when attempting to control more than one servo on the same chip. After some research, we were able to attach a 220µF capacitor in order to act as a reserve for extra power needed at different times for the servos. Additionally, the capacitor was able to help filter out transient signals that may have been present in the control line.

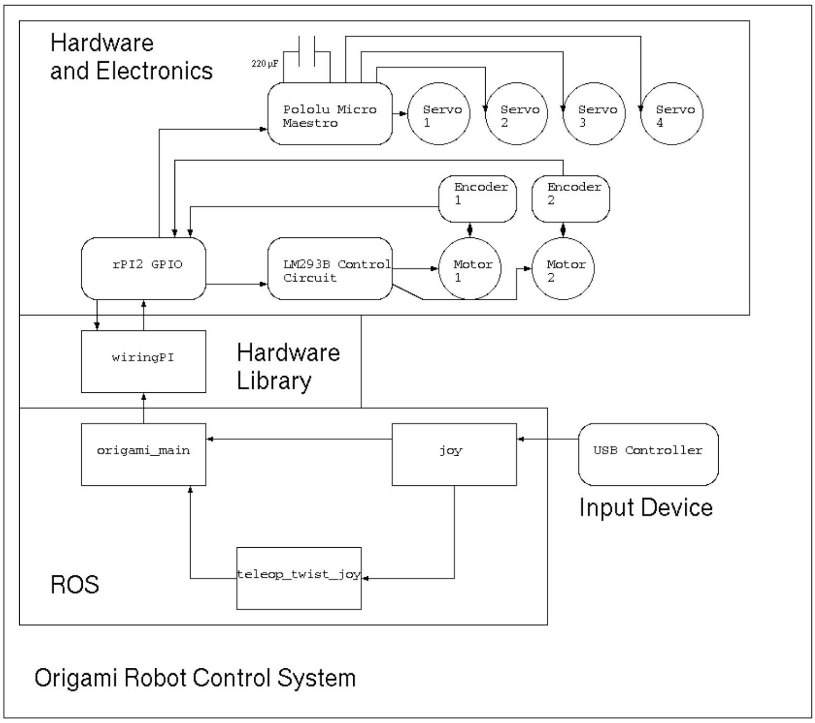


Fig. 5. Origami Control System

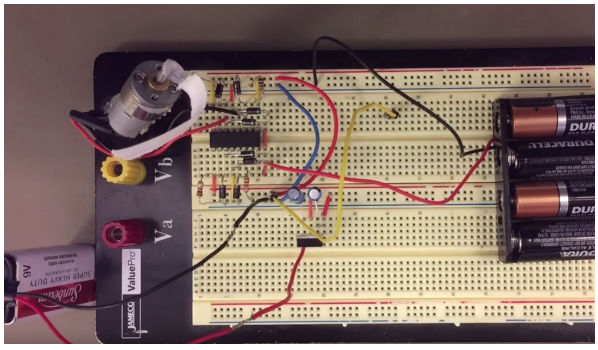


Fig. 6. Testing on a breadboard

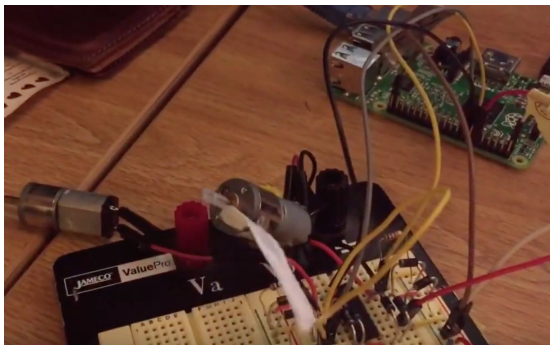


Fig. 7. PWM'd motors

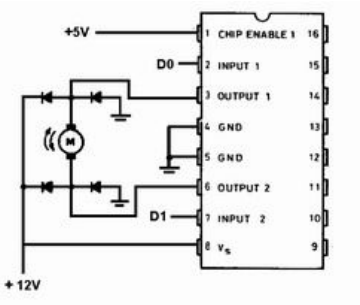


Fig. 8. Basis of the motor control circuit

Fig. 9. Motor Encoders

C. Software Design

For our robot, we opted to program it using ROS, the Robot Operating System, which is a software framework that runs as middleware between user-defined programs, installable packages, and hardware that makes the robot run. The governing principle that makes ROS useful is its ability to run software in a distributed, nodal, way. Each piece of software that runs on a ROS system is called a "node". These nodes exchange messages with one another by publishing and subscribing to different topics all routed by a central node called the "roscore". The advantage of using ROS is that it is widely supported and already contains a lot of software packages for different robot functions. Our software consisted of a single ROS package written with a single primary node, called *origami_main*, and provides a series of different functions to make the origami robot work. It is typically ran via a ROS "launch file" which allows for the running of multiple nodes along with a roscore by only entering a single command. The launch file launches the *origami_main* node, which is responsible for all real hardware control, the *joy* node, which provides an easy-to-use interface for a USB joystick connected to the Raspberry Pi, and the *teleop_twist_joy* node, which provides ROS message-level translation of any standard joystick input to a speed command message called a *Twist* that may be interpreted by the *origami_main* node. Additionally, the launch file specifies a configuration file to be included at launch. This configuration file contains variables that are loaded at runtime for specifying the rate at which odometry is calculated, how many encoder ticks there are per meter, and the width of the base in meters. Odometry is not included within *origami_main*, but instead resides in a separate, untested, node. The configuration file also contains information supplies to the *joy* and *teleop_twist_joy* nodes about which joystick to use, and some linear scaling. Within the origami_main node, several key functions are performed. The foremost purpose of this node is to provide an interface with the Raspberry Pi's general purpose I/O library, WiringPI. The Raspberry Pi provides many different pins which can be controlled and configured using this library. The pins may be made to generate pulse-width modulated (PWM) signals for variable speed control, signal as HIGH or LOW, or read in analog or digital signals.

For the purposes of our project, pins were allocated for speed control of the left and right motors, reading the encoder values, and signaling the Pololu Micro Maestro to transform the wheels. The speed control pins were initialized using the soft PWM interface provided by WiringPI, which allowed software generated PWM signals. This was necessary as PWM signals are traditionally generated using hardware timers, but as the Raspberry Pi only has one hardware PWM pin normally used for audio generation, the software solution was used. To count the encoder ticks, four pins were set up to read a digital input signal. The encoder was of the rotary type and had four different states to indicate four different positions of the encoder while the motor spun around. These states of the pins were the following set of tuples: {[0,0], [0,1], [1,0], [1,1]}. As we did not need to determine the exact location of the motor shaft at any one point in time, and the odometry node was not entirely flushed out, encoder ticks were counted by counting the number of different signals coming in, then for every four changes, increasing the tick count and resetting the change counter. This technique allowed for us to count full rotations of the motor quickly

and cheaply.In order to transform the wheels using the servos and Pololu Micro Maestro, a signal is sent from the Raspberry Pi to the Micro Maestro. This signal is a short pulse that lets the Maestro know that it is time to change the state of the servos. The final function of the origami_main node is to receive "Twist" commands and turn them into motor velocities. The node listens on the ROS topic "/cmd_vel" for an input generated by the teleop_twist_joy node. This Twist specifies a linear and angular velocity that is interpreted by the node and scaled appropriately to turn into PWM commands for both the left and right motors.

Additionally, on the Raspberry Pi, there is an odometry node. Due to time constraints, this node remained untested; but was based off of odometry calculations provided by the ROS package "differential_drive". An initial attempt to use the differential_drive package was made, but as it is a legacy package, support for it has mostly ceased and its use resulted in some transform errors. Therefore, the odometry code was examined and re-written in C++ for the purposes of this project. Odometry for a differential drive robot may be mathematically described in the following way [4]:

Let θ_L, θ_R be the measured rotation of the left and right wheels, respectively.

Then

$$\theta = \frac{(\theta_R - \theta_L) r_W}{b} + \theta_0$$

$$x = x_0 + \frac{(\theta_R + \theta_L) b}{(\theta_R - \theta_L) 2} [\sin(\theta) - \sin(\theta_0)]$$

$$y = y_0 - \frac{(\theta_R + \theta_L) b}{(\theta_R - \theta_L) 2} [\cos(\theta) - \cos(\theta_0)]$$

When $\theta_R - \theta_L \approx 0$ the robot does not turn, but follows a straight-line path

$$x = x_0 + \frac{(\theta_R + \theta_L) r_W}{2} \cdot \cos(\theta_0)$$

$$y = y_0 + \frac{(\theta_R + \theta_L) r_W}{2} \cdot \sin(\theta_0)$$

Lastly, there is the software that runs on the Pololu Micro Maestro to toggle the state of the servos. The Micro Maestro supports a stack-dependent scripting language for control of up to six servos. In order to provide the easiest possible control of the servos from the Raspberry Pi, the majority of the work was offloaded onto the Micro Maestro. The code begins by placing a state variable for whichever current state the wheels are in into the stack. Then, it enters into a loop, continually checking the state of a pin on the device that was labeled for input. Once the pin is supplied with a HIGH signal from the Raspberry Pi, the state variable is checked, and the servos begin a sequence to actuate the wheels into an alternate state. If the wheels were previously closed, they will open, and vice versa. After the process is complete, the program changes the state variable on the stack to be in accordance with its new state, and returns to the initial loop. This process allows us to control the servos using only one simple signal from the Raspberry Pi.

Software for this project is available on GitHub here: https://github.com/lucbettaieb/origami_bot

III. STRUCTURE ANALYSIS

Fig. 10 and 11 shows the basic components of the wheel, and rotation direction and angles are assigned. To analyse the structure of the origami wheel and the deformation mechanism, all the points needed also have been named. In Fig. 10, we assume it as the start status before shrinkage. Then, take the linkage BB' and FF' as examples:

$$BB' = \frac{OB \times \sin\alpha - OB' \times \sin\beta}{\sin\gamma} \qquad FF' = \frac{OF \times \sin\theta - OF' \times \sin\eta}{\sin\phi}$$

After testing and comparing the deformation function, we decided to set the curved linkages in opposite direction on the two sides of a wheel, which has the best deformation results. Then, servos rotated the plate, and linkages dreg 8 components of the wheels shrink into the shape of Fig. 11. The angle of ∠BOB', which equals to **α**-**β**, changed into **α**1+**β**1(∠B1OB1'; and ∠FOF', which is **θ**-***η***, turned into **θ**1+***η***1. Besides, the maximum rotation speed is approximately 1.54 r/s.

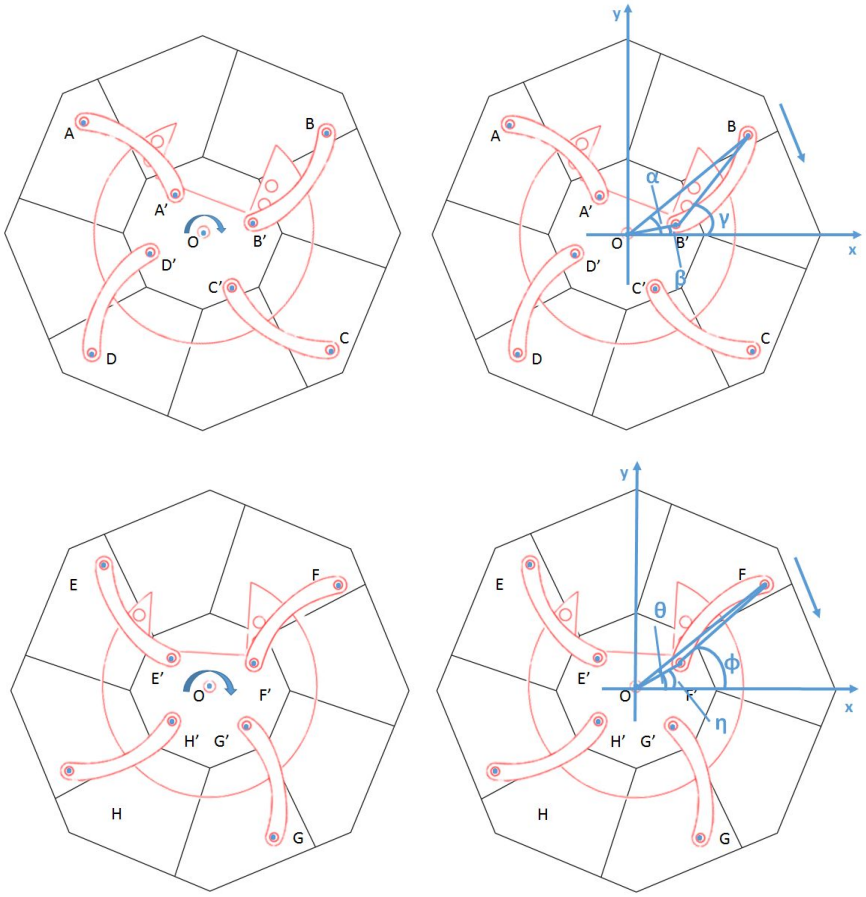


Fig. 10. Octagon-shape wheel

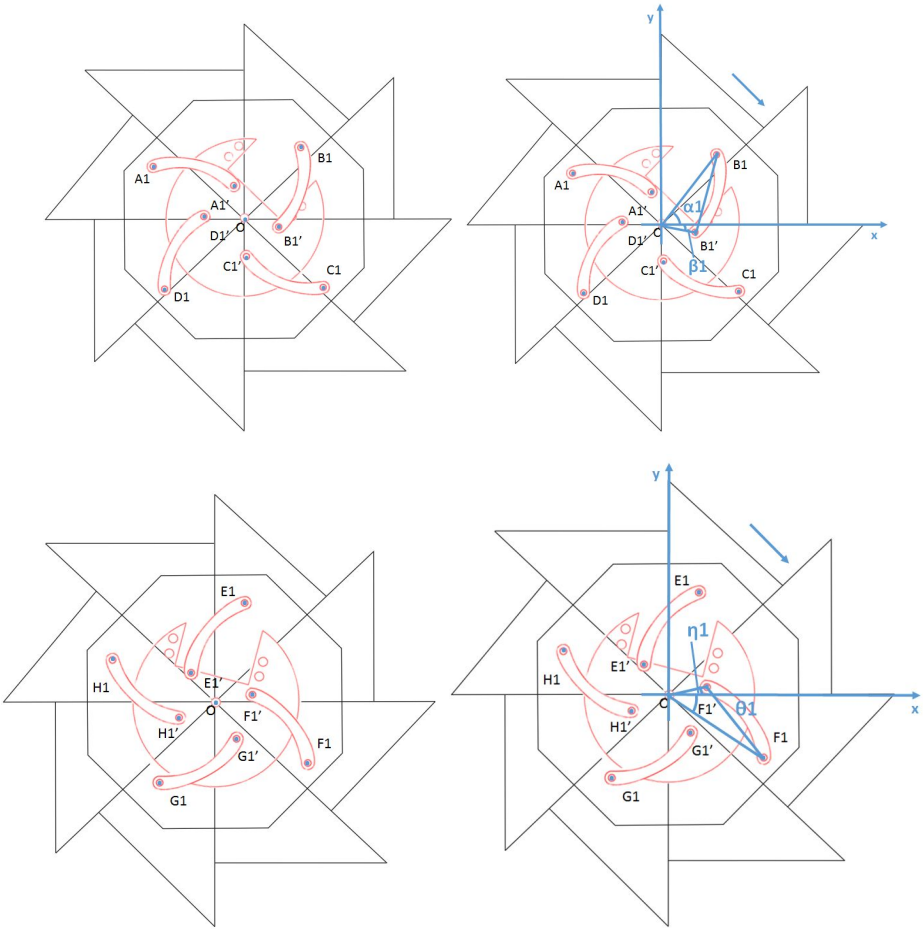


Fig. 11. Spiky-shaped wheel

IV. FABRICATION

The origami wheel is consisted by eight parts hand-folded from square transparency film with 6 inch width and 0.016 inch thickness. Sixteen 1.15 inch long curved linkages, four 1.75 in radius plates (Fig. 4) and the chassis were laser-cut from 0.25 inch thick birch plywood. Also, size #2-56 nuts, screws and washers were used to connect plates and linkages, and size #10-32 threaded rods were chosen as shafts. Brass motor shaft coupling connectors were applied to connect wheel's shaft and motor shaft. The weight of the wheel (plus servo, gears, screws, nails, linkages, etc) is 230g.

V. RESULTS

In this experiment, we have tested the forward movement with octagon-shape wheels on a smooth and flat ground and wheels deformation. If the micro-servos are used, forward movement works well. Slippage doesn't occur when the wheels move forward. The structures folded by plastic material can afford the load of the chassis and the weight of servos. The shape of the wheels won't change if we don't give the deformation command which means the gears lock very tightly. Also, the two caster wheels installed in the rear move smoothly to support the chassis.

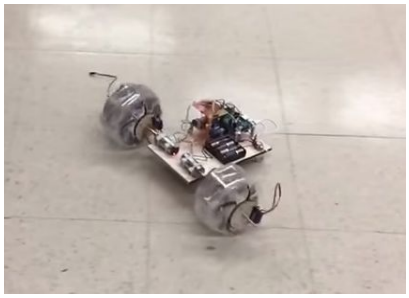
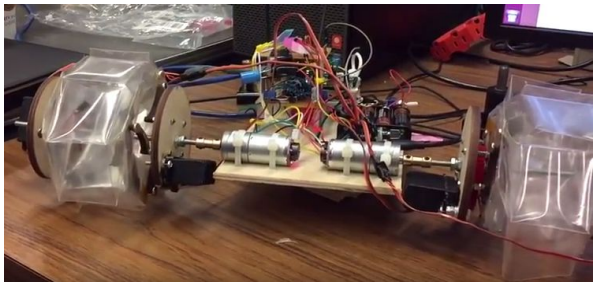
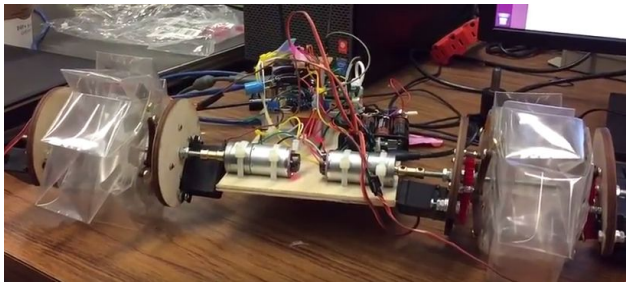


Fig.12 Forward movement with octagon-shape wheels on a smooth and flat ground

We tested the wheels deformation when it is stationary. The right origami wheel works well when the command is sent. The linkages installed in opposite directions are controlled by two servos also turned in the opposite directions. The forces on the wheels expand and shrink the wheels in a good way. The left one, however, was not as effective as the right wheel. This was likely due to inconsistencies in our manufacturing procedure.



(a) Wheels expansion



(b) Wheels shrink

Fig.13 wheels deformation

VI. BUDGET

Part	**Quantity**	**Price**
Raspberry Pi 2	1	$35.32
PETG	2	$14.44
Construction Paper	2	$14.28
Transparency Film	1	$10.91
VHB Tape	1	$7.85
Servo	1	$8
Encoder Motor	2	$20.7
Gear Set	2	$5.62
Brads	1	$6.11
USB Battery	1	$6
Casters	1	$6.65
Wood Board	1	$9.75
Total		$145.63

VII. CONCLUSION and DISCUSSION

As a group, we learned a lot while working on this project. Our prototype design of the wheels, control system, and the robot as a whole allowed for us to to learn about both its positive and negative aspects. Key features we hope to implement on the next revision are 3D-printed wheels that can support more load and be transformed more smoothly, a custom made or consumer off-the-shelf printed circuit board containing the main motor control circuit, better motors that will be able to supply enough torque to turn the wheels efficiently, and a new power system with a better battery.

As this vehicle's purpose is to be a functioning prototype for demonstrating technology that may be applied to a micro-scale planetary rover someday, we believe that the research performed throughout the course of this project has been a valuable step in the right direction. After addressing the majority of the problems that presented themselves throughout the course of this project, the robot may be tested on a variety of different terrains. These terrains may include mud, sand, water, dirt, gravel, and more. We hope to soon fully realize a robot that can traverse all these terrains, and more.

ACKNOWLEDGEMENTS

This work was supported by Department of Mechanical and Aerospace Engineering in Case Western Reserve University and advised by Dr. Kiju Lee in Case Western Reserve University. Additional thanks to George Daher, who assisted with the design of the motor control circuit.

REFERENCES

[1] Dae-Young Lee, Ji-Suk Kim, Sa-Reum Kim, Je-Sung Koh, Kyu-Jin Cho, the Deformable Wheel Robot using Magic-ball Origami Structure, in: Proceedings of the ASME 2013 International Design Engineering Technical Conference & Computers and Information in Engineering Conference(IDETC/CIE), Portland, Oregon, 2013.

[2] Dae-Young Lee, Ji-Suk Kim, Jae-Jun Park, Sa-Reum Kim, Kyu-Jin Cho, Fabrication of Origami Wheel using Pattern Embedded Fabric and its Application to a Deformable Mobile Robot, in: IEEE International Conference on Robotics & Automation(ICRA), Hong Kong, China, 2014.

[3] Dae-Young Lee, Je-Sung Koh, Kyu-Jin Cho, Ji-Suk Kim, Seung-Won Kim, Kyu-Jin Cho, Deformable-wheel Robot based on Soft Material, in: International Journal of Precision Engineering and Manufacturing Vol. 14, No. 8, pp. 1439-1445.

[4] G. W. Lucas, "Contents," A Compendium to Calculations Useful for Robotics, 16-Jan-2006. [Online]. Available at: http://rossum.sourceforge.net/papers/calculationsforrobotics/compendiumforkinematics.htm. [Accessed: 28-Apr-2016].

[5] Samuel M. Felton, Dae-Young Lee, Kyu-Jin Cho, Robert J. Wood, A Passive, Origami-Inspired, Continuously Variable Transmission, in: IEEE International Conference on Robotics & Automation(ICRA), Hong Kong, China, 2014.

[6] Jeremy Shafer, Origami 3-D Transforming Ninja Star Designed by Ray Bolt [Online] https://www.youtube.com/watch?v=yXGkLnUoQe4

[7] Jo Nakashima, Origami Origami Magic Ball (Dragon's Egg by Yuri Shumakov)[Online] https://www.youtube.com/watch?v=VgXwSdJNks8

[8] Jo Nakashima, Origami Flexible (Jorge Pardo) [Online] https://www.youtube.com/watch?v=knMEBSXM6WU