Rollbot: a Spherical Robot Driven by a Single Actuator

Jingxian Wang[1] and Michael Rubenstein[1]

***Abstract*— Here we present Rollbot, the first spherical robot capable of controllably maneuvering on 2D plane with a single actuator. Rollbot rolls on the ground in circular pattern and controls its motion by changing the curvature of the trajectory through accelerating and decelerating its single motor and attached mass. We present the theoretical analysis, design, and control of Rollbot, and demonstrate its ability to move in a controllable circular pattern and follow waypoints.**

I. Introduction

Underactuated robots has attracted researchers' attention in recent years due to their advantages such as energy savings, material savings, space savings, etc. [1]. While underactuated systems like inverted pendulum [2, 3], underactuated hands [4, 5], and underactuated wrists [6, 7] have been extensively studied, few mobile robots have adopted the idea of underactuation due to the difficulty of designing movement mechanism that can utilize non-holonomic constraints and precisely controlling underactuated systems.

Until now, only a handful of mobile robots can maneuver in 2D or 3D space with a single actuator. Robots in [8–10] used compliant mechanisms or one-way bearing to allow a single actuator to achieve multiple functions or have multiple mode of operation, effectively 'multiplexing' the actuator to move in space. Walking robot in [11] controls its movement by controlling left and right leg's step size. Robots in [12–15] and drones in [16, 17] move by rotating their body and accelerate forward at the appropriate time. Among all these works, only [15] proposed a wheeled robot, and it is also the only ground robot that do not move in discrete steps.

Another key inspiration of our work is spherical robots [18] studied for their structural simplicity and the inherent protective nature of their spherical shells that shields the internal mechanisms and electronics from hostile environments. Since the pioneering Rollo robot [19], three major types of rolling robot driven mechanisms has been proposed [18], including barycentric type [20, 21] that moves masses attached to pendulums or tracks around the robot, conservation of the angular momentum (CoAM) type [22] that uses reaction wheels or gyroscopes, and shell deformation type [23] that changes the shape of the shell. Interestingly, no existing spherical robot of any driving type have only one actuator to authors' knowledge, and even the authors of review paper [18] claimed that 'For a robot to be able to move in more than one direction, at least two DoFs are needed.'

[1]Jingxian Wang and Michael Rubenstein are with the Center for Robotics and Biosystems, Northwestern University, Evanston, IL 60601, USA jingxianwang2026@u.northwestern.edu, rubenstein@northwestern.edu

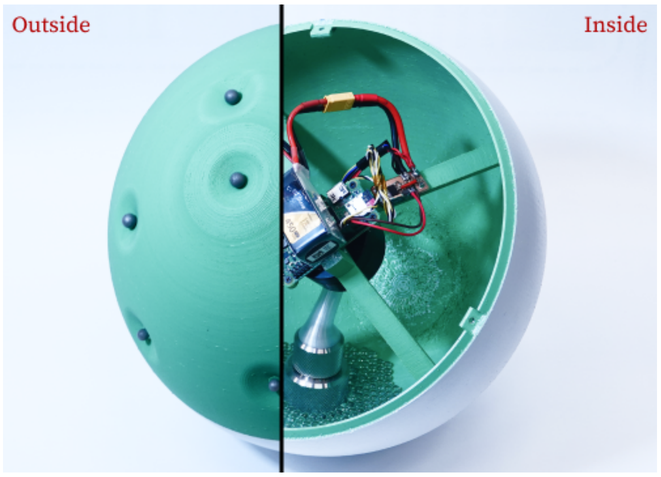

Fig. 1. Photo of inside and outside of Rollbot. Rollbot has a outer diameter of 24 cm and weighs 1.2 kg.

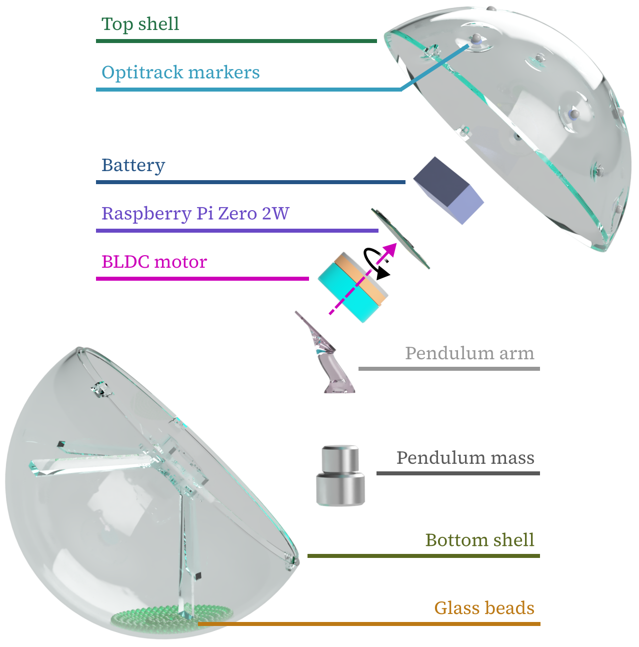

Fig. 2. An exploded view of Rollbot. The only actuator is the BLDC motor at the center which can rotate around the purple axis.

The lack of research in single actuator spherical robots motivates us to create Rollbot, a barycentric type spherical robot driven by a pendulum attached to a single motor. Rollbot can move on 2D plane by exploiting the non-holonomic rolling constraint.

II. Robot Design

Rollbot, as shown in Fig. 1 and 2, sets out to test the feasibility of utilizing non-holonomic rolling constraint to achieve 2D motion using only a single motor actuator. RollBot is composed of a spherical shell and a pendulum connected to the shell through a motor. As the motor rotates the pendulum, The shell will also roll on the ground, and we can control the motion of Rollbot by changing the rotating speed of the motor. For concision, we will refer to the spherical shell as the shell, the pendulum mass as the mass, and the motor's spinning speed as driving speed in future discussions.

A. Dynamics of Rollbot

Rollbot can be considered as a spherical rolling robot driven by a internal mass moving on a circular trajectory in the shell's reference frame. In this section, we will derive the dynamics of a general spherical rolling robot driven by the movement of internal point masses and apply the result to Rollbot. We will start with a simple case where a spherical symmetric shell is driven by one internal point mass as illustrated in Fig. 3. We also assume that the robot will not lose contact with the ground and will not slip.

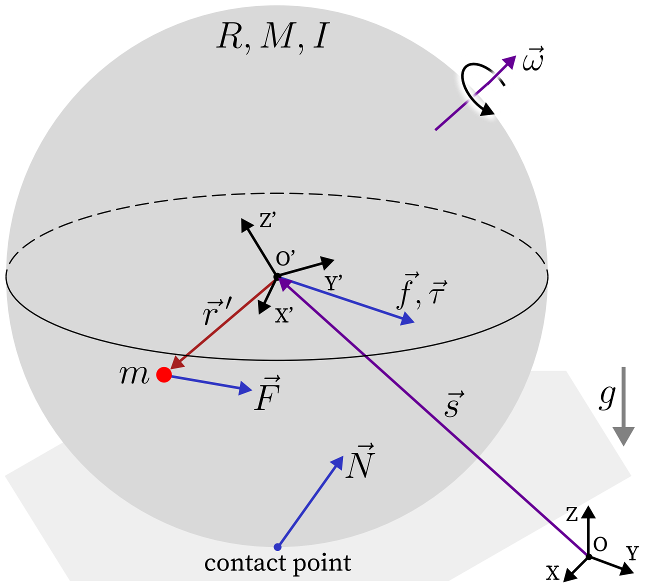

Fig. 3. Illustration of relevant quantities. O-XYZ is the ground reference frame G, O'-X'Y'Z' is the shell's body reference frame B. $\vec{s}$ is the position of the center of the shell in G, $\vec{\omega}$ is the angular velocity of the shell in G, and $\vec{r}\,'$ is the position of the mass in B. R, M, I are the radius, mass, and the moment of inertia of the shell respectively, and m is the mass of the point mass. $\vec{f}, \vec{\tau}$ are the externally applied force and torque on the shell with respect to O', $\vec{N}$ is the sum of normal and friction force ground applies on the shell, and $\vec{F}$ is the force shell applies on the point mass.

The vector from the center of shell to the point mass $O'm$ in ground frame, $\vec{r}$, could be described using the rotation matrix $\mathbf{T}$ from body frame to ground frame as

$$\vec{r} = \mathbf{T} \cdot \vec{r}\,' \tag{1}$$

We can take its first and second derivative and obtain

$$\dot{\vec{r}} = \vec{\omega} \times (\mathbf{T} \cdot \vec{r}\,') + \mathbf{T} \cdot \dot{\vec{r}}\,' \tag{2}$$

$$\begin{aligned}\ddot{\vec{r}} =& \dot{\vec{\omega}} \times (\mathbf{T} \cdot \vec{r}\,') \\ &+ (\vec{\omega} \times (\vec{\omega} \times (\mathbf{T} \cdot \vec{r}\,')) + 2\vec{\omega} \times (\mathbf{T} \cdot \dot{\vec{r}}\,') + \mathbf{T} \cdot \ddot{\vec{r}}\,') \\ =& \dot{\vec{\omega}} \times (\mathbf{T} \cdot \vec{r}\,') + \vec{h}(\vec{\omega}, \mathbf{T}, \vec{r}\,'(t))\end{aligned} \tag{3}$$

where $\dot{a}$ is the derivative of a over time. Notice that term $\vec{h}$ is the inertial force applied on the point mass in the non-accelerating co-rotating frame and does not involve acceleration of the shell.

Newton's second law and angular momentum theorem give us the equation of motion of the point mass and the shell.

$$m(\ddot{\vec{s}} + \ddot{\vec{r}}) = \vec{F} - mg\hat{z} \tag{4}$$

$$M\ddot{\vec{s}} = -\vec{F} + \vec{N} - Mg\hat{z} + \vec{f} \tag{5}$$

$$I\dot{\vec{\omega}} = -(\vec{r} - \vec{s}) \times \vec{F} + (-R\hat{z}) \times \vec{N} + \vec{\tau} \tag{6}$$

We could cancel out $\vec{F}$ and $\vec{N}$ by merging the three equations

$$\vec{u} \times (\text{Eqn.4}) + R\hat{z} \times (\text{Eqn.5}) + (\text{Eqn.6}) \tag{7}$$

where $\vec{u} = R\hat{z} + \vec{r}$ is the vector from contact point to the point mass and obtain

$$\begin{aligned}&\vec{u} \times (m(\ddot{\vec{s}} + \ddot{\vec{r}})) + (R\hat{z}) \times (M\ddot{\vec{s}}) + I\dot{\vec{\omega}} \\ &= \vec{u} \times (-mg\hat{z}) + (R\hat{z}) \times \vec{f} + \vec{\tau}\end{aligned} \tag{8}$$

Then we can apply the no slipping constraint

$$\dot{\vec{s}} = \vec{\omega} \times (R\hat{z}) \tag{9}$$

and (1), (2), (3) to (8) and obtain the complete equation of motion of the shell as

$$\begin{aligned}&(m(\vec{u} \cdot \vec{u}\mathbf{I}_3 - \vec{u}\vec{u}) + MR^2(\mathbf{I}_3 - \hat{z}\hat{z}) + I\,\mathbf{I}_3) \cdot \dot{\vec{\omega}} \\ &= m\vec{u} \times (-\vec{h}(\vec{\omega}, \mathbf{T}, \vec{r}\,'(t)) - g\hat{z}) + R\hat{z} \times \vec{f} + \vec{\tau}\end{aligned} \tag{10}$$

where $\mathbf{I}_3$ is the 3rd order identity matrix, and $\vec{p}\vec{q}$ is the outer product of $\vec{p}$ and $\vec{q}$.

The physical meaning of (10) is clear: the left hand side is the moment of inertia matrix of the whole robot with respect to the contact point at this moment applied on the angular acceleration, and the right hand side is the torque applied on the robot with respect to the contact point contributed by the point mass's inertia in the non-accelerating co-rotating frame, the point mass's gravity, and the external force and torque applied on the shell.

We can also write out the supporting and friction force

$$\vec{N} = (M + m)\ddot{\vec{s}} + m\ddot{\vec{r}} + (M + m)g\hat{z} - \vec{f} \tag{11}$$

The condition for the robot to not jump and not slip could be expressed as

$$\vec{N}\cdot\hat{z} > 0 \tag{12}$$

$$\frac{\vec{N}\cdot\hat{z}}{\|\vec{N}\|} \geq \frac{1}{\sqrt{1+\mu^2}} \tag{13}$$

where μ is the friction coefficient between the ground and the shell.

This equation of motion we derived can be extended to handle more general cases. Adding a summation to all terms related to the point mass allows it to deal with multiple point masses.

$$\left(\sum_q m_q(\vec{u}_q\cdot\vec{u}_q\,\mathbf{I}_3 - \vec{u}_q\vec{u}_q) + MR^2(\mathbf{I}_3 - \hat{z}\hat{z}) + I\,\mathbf{I}_3\right)\cdot\dot{\vec{\omega}} = \sum_q m_q\vec{u}_q \times (-\vec{h}(\vec{\omega},\mathbf{T},\vec{r}'_q(t)) - g\hat{z}) + R\hat{z}\times\vec{f} + \vec{\tau} \tag{14}$$

where m_q is the mass of qth point mass, $\vec{r}'_q$ is the position of qth point mass in B, and $\vec{u}_q$ is the vector from the contact point to qth point mass. Furthermore, by distributing extra point masses fixed to the shell or other point masses, (14) can also deal with cases involving non-symmetrical mass distribution of shell or general solid bodies moving inside the robot.

The result we obtained is equivalent to what is proposed in [24], but the derivation is more straight forward and has clearer physical meaning.

When applying the model to Rollbot's case, a good approximation would be directly use (10) and let

$$\vec{r}' = r_0(\sin\phi\cos\theta(t)\,\hat{x} + \sin\phi\sin\theta(t)\,\hat{y} - \cos\phi\,\hat{z}) \tag{15}$$

$$\vec{\tau} = -k_0\vec{\omega} \tag{16}$$

$$\vec{f} = \mathbf{0} \tag{17}$$

$$M = M_s + m_b \tag{18}$$

where $\theta(t)$ is the rotating angle of the pendulum mass around Z' axis, and k_0 is an an experimentally determined damping constant. Such simplification is valid when

- The shell has moment of inertia matrix in the form of $I\,\mathbf{I}_3$.
- The pendulum mass's size is much smaller than $|\vec{r}'|$ and the radius of the shell R, so we can simplify the pendulum mass to a point mass.
- The damping on shell's motion is small and have small impact on the behavior of Rollbot.

and we will show that our system satisfy these requirements in section II-D.

B. Quasi-Static State of Rollbot

In our experiments, we are slowly accelerating and decelerating the pendulum mass, so Rollbot is close to the quasi-static state where the driving speed $\theta'(t) = \omega_0$ is a constant. In the quasi-static state, Rollbot will revolve around a vertical axis l, and in the frame revolving together with the Rollbot, Rollbot's movement is always the same, as shown in Fig. 4.

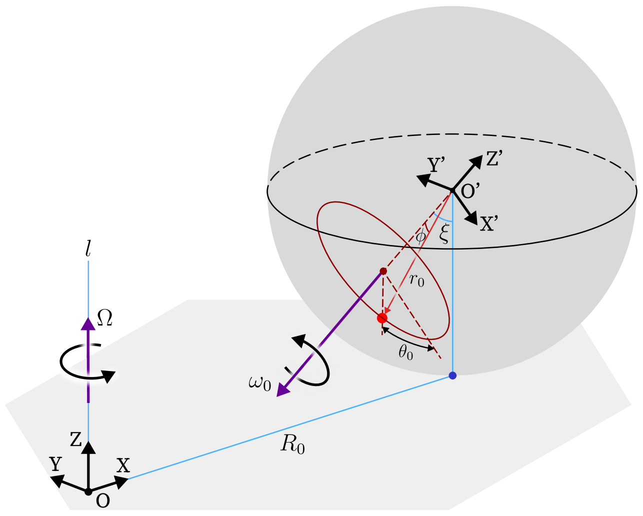

Fig. 4. Illustration of quasi-static state at $t = 0$. O-XYZ is the non-moving ground reference frame G with origin at the intersection of revolving axis l and ground plane, and X axis pointing towards the contact point. O'-X'Y'Z' is the shell's body reference frame B with Y' axis parallel to Y' axis and Z' aligned with the rotation axis of the motor and on X-Z plane, and the angle between Z and Z' is ξ. The shell is revolving around l at angular velocity $\Omega\hat{z}$, plus the shell itself is also spinning at angular velocity $\vec{\omega}_s = -\omega_0\hat{z}'$ in quasi-static state. The shell's revolving radius is R_0.

Because the whole system is revolving around l at constant angular velocity Ω, we have

$$\vec{u}(t) = \mathbf{T}_r(t)\cdot\vec{u}(0) \tag{19}$$

$$\vec{\omega}(t) = \mathbf{T}_r(t)\cdot\vec{\omega}(0) \tag{20}$$

where $\mathbf{T}_r(t)$ is the rotation matrix defined as

$$\mathbf{T}_r(t) = \begin{bmatrix}\cos(\Omega t) & -\sin(\Omega t) & 0\\ \sin(\Omega t) & \cos(\Omega t) & 0\\ 0 & 0 & 1\end{bmatrix} \tag{21}$$

Using this property, we can then express (10) using only variables at $t = 0$, and subsequently reduce it into three equations about Ω, θ_0, and ξ.

$$m\vec{u}\times(-g\vec{z} + \Omega^2(R_0\hat{x} + \vec{u} - (\vec{u}\cdot\hat{z})\hat{z}) + M\Omega^2R_0R\hat{y} - k_0(-\omega_0\hat{z}' + \Omega\hat{z}) + I\Omega\omega_0\sin\xi\hat{y} = \mathbf{0} \tag{22}$$

where $R_0 = \omega_0 R\sin\xi/\Omega$ and $\hat{z}' = \sin\xi\hat{x} + \cos\xi\hat{z}$. The quasi-static state of Rollbot at different driving speed ω_0 can then be obtained by numerically solving (22). The solution with Rollbot's actual physical parameters is shown in Fig. 5.

Fig. 5 shows that the revolving radius R_0 starts from $R\tan\phi$ when $\omega_0 = 0$, and gradually increases with ω_0, hinting that we could control the trajectory of Rollbot by changing the driving speed. ξ starts from ϕ when $\omega_0 = 0$ and gradually approaches $\pi/2$, meaning that when Rollbot is almost moving straight forward when rolling fast. θ_0 is close to 0, meaning that the pendulum mass is always near the bottom of the shell when driving speed is smaller than 3π.

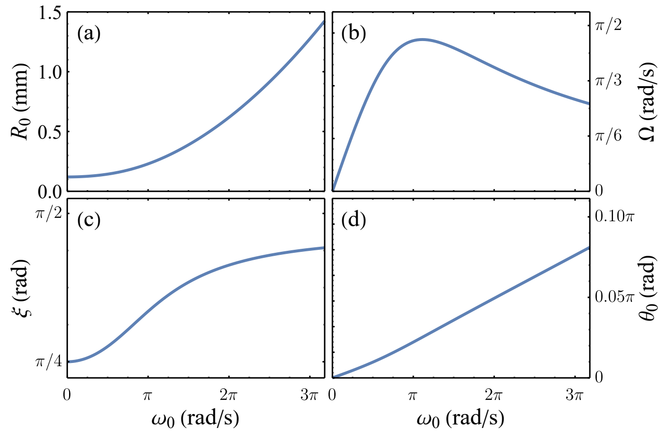

Fig. 5. Plots of key parameters of quasi-static state versus driving speed ω_0. (a) to (d) show the trends of revolving radius R_0, revolving angular velocity Ω, tilting angle of rotating axis of the motor ξ and rotating angle of pendulum mass θ_0 respectively.

Furthermore, we could also discover that as long as damping coefficient k_0 is not exceedingly large, it has little impact on the result. This means that we can make assumption (16) for convenience of computation without influencing the result significantly.

Aside from the state shown above, there is also another set of 'fast revolving state', where θ_0 is close to π and Rollbot revolves in the opposite direction at a much smaller radius. This set of state can rarely be observed in experiments.

C. Perturbation around Quasi-Static State

Perturbation around the quasi-static state tells us how stable the quasi-static state is and the response of Rollbot if we deviate from the state, either during acceleration and deceleration or due to external disturbances. In further derivation, we will use a^p to represent a perturbed physical quantity a and δa to represent the perturbation of a, i.e. $\delta a = a^p - a$. We will only consider the first order perturbation in the following discussion.

We can apply a perturbation to the rotation matrix $\mathbf{T}$ and have

$$\mathbf{T}^p(t) = (1 + [\mathbf{T}_r(t) \cdot \vec{\alpha}(t)]) \cdot \mathbf{T}(t) \tag{23}$$

where $\vec{\alpha}$ is a small perturbation vector, and $[\vec{\alpha}]$ is the skew-symmetric form of $\vec{\alpha}$. According to the definition, we have

$$\begin{aligned} \delta\vec{u} = \delta\vec{r} &= (\mathbf{T}_r \cdot \vec{\alpha}) \times \vec{r} \\ &= \mathbf{T}_r \cdot (\vec{\alpha} \times \vec{r}(0)) \end{aligned} \tag{24}$$

$$\begin{aligned} \delta\vec{\omega} &= (\mathbf{T}_r \cdot \vec{\alpha}) \times \vec{\omega} + d_t(\mathbf{T}_r \cdot \vec{\alpha}) \\ &= \mathbf{T}_r \cdot (\vec{\alpha} \times \vec{\omega}_s(0) + \dot{\vec{\alpha}}) \end{aligned} \tag{25}$$

Put (24) and (25) into (9) and we have

$$\delta\dot{\vec{s}} = \mathbf{T}_r \cdot ((\vec{\alpha} \times \vec{\omega}_s(0) + \dot{\vec{\alpha}}) \times (R\hat{z})) \tag{26}$$

Take perturbation of (8), left multiply by $\mathbf{T}_r^{-1}$ and use (26) then we have

$$\begin{aligned} & m(\vec{\alpha} \times \vec{r}(0)) \times (\ddot{\vec{u}}(0) + \ddot{\vec{s}}(0)) \\ & + m\vec{u}(0) \times \Omega^2(\vec{\alpha} \times \vec{r}(0) - ((\vec{\alpha} \times \vec{r}(0)) \cdot \hat{z})\hat{z}) \\ & + m\vec{u}(0) \times (2\Omega\hat{z} \times (\dot{\vec{\alpha}} \times \vec{r}(0)) + \ddot{\vec{\alpha}} \times \vec{r}(0)) \\ & + (m\vec{u}(0) + MR\hat{z}) \times ((\dot{\vec{\alpha}} \times \vec{\omega}_s(0) + \ddot{\vec{\alpha}}) \times (R\hat{z})) \\ & + I(\dot{\vec{\alpha}} \times \vec{\omega}_s(0) + \ddot{\vec{\alpha}}) \\ & + mg(\vec{\alpha} \times \vec{r}(0)) \times \hat{z} \\ & + k_0(\vec{\alpha} \times \vec{\omega}_s(0) + \dot{\vec{\alpha}}) = 0 \end{aligned} \tag{27}$$

Notice that this is an second order ordinary differential equation group of $\vec{\alpha}(t)$ and the parameters are not dependent on time, we could plug in the values obtained in section II-B and obtain an equation in the following form

$$\begin{pmatrix} \dot{\vec{\alpha}} \\ \ddot{\vec{\alpha}} \end{pmatrix} = \begin{pmatrix} 0 & \mathbf{I}_3 \\ \mathbf{A}_{3\times3} & \mathbf{B}_{3\times3} \end{pmatrix} \cdot \begin{pmatrix} \vec{\alpha} \\ \dot{\vec{\alpha}} \end{pmatrix} \tag{28}$$

The recovery behavior of Rollbot under perturbation is characterized by eigenvalues of the matrix in (28).

In addition, notice that the system will be identical if we make a perturbation and let Rollbot revolve around l by a certain angle, we should remove the eigenvalue and eigenvector corresponding to this mode of movement when analyzing the stability. Finally, we can conclude that if all but one eigenvalues have real part smaller than 0, then Rollbot is stable under perturbation and the characteristic time of recovery τ is determined by the eigenvalue λ_2 with second smallest real part.

$$\tau \approx -\frac{1}{Re(\lambda_2)} \tag{29}$$

Plot of eigenvalues' distribution under different ω_0 with Rollbot's actual physical parameters is shown in Fig. 6, showing that with our choise of parameters, Rollbot is able to recover from perturbations with characteristic time of about 7 s. This is consistent with the result in direct numerical simulation as well as our experiments.

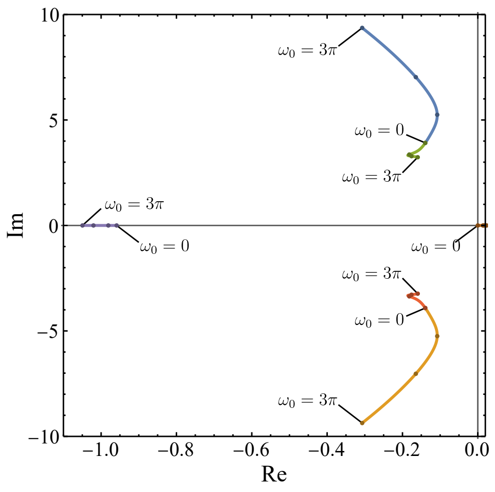

Fig. 6. Plot of eigenvalues' trajectory for $\omega_0 \in [0, 3\pi]$.

As far as the authors know, this is the first time quasi-static state and perturbation behavior are analyzed for spherical robots.

D. Hardware Design

When designing the hardware, we want it to satisfy the following criterion:

- Rollbot should be as close to our theoretical model as possible, i.e. satisfying the criteria given in section II-A.
- Rollbot should be close to quasi-static state even when accelerating/decelerating.
- Rollbot's trajectory should change significantly as we accelerate/decelerate the motor.

The first criteria require us to design the Rollbot shell's center of mass near the shell's geometric center, a moment of inertia in the form of $I_0\mathbf{I}_3$, and pendulum mass small enough to be considered as a point mass, and the second requires Rollbot to have fast recovery from transient state, and third requires revolving radius R_0 to change significantly when we change the driving speed ω_0.

We designed the diameter of Rollbot's shell to be 24 cm, which is a few times larger than the scale of the battery and the motor, and concentrated the mass at the spherical surface of the shell. This make sure that the moment of inertia is dominated by the surface of the shell instead of its internal structures, keeping the moment of inertia in the form of $I_0\mathbf{I}_3$. Furthermore, we also carefully positioned the motor and the battery to keep the center of mass near the the shell's geometric center. The pendulum mass is made of stainless steel and is small comparing to the distance between it and the center of the shell as well as the size of the shell. We put $m_b = 40\,\text{g}$ of glass beads in the shell so the collision between beads can introduce energy dissipation and create damping. Because the total mass of the beads is much smaller than the pendulum mass or the shell, and beads are always staying near the bottom of the shell, we can simplify the beads' effect to an addition of mass to the shell plus a linear damping given by (16).

With the analysis in section II-C, we find out that the system is more stable when r_0, m, ϕ, and k_0 are larger. As such, we placed the pendulum mass as close to the shell as possible, and introduced the beads inside the shell. Experimentally, comparing to the case where no beads are added, Rollbot's quasi-static dynamics is almost the same while exhibiting much less oscillation and much faster recovery after acceleration and deceleration.

With the analysis in section II-B, we could notice that the revolving radius R_0 changes more significantly with ω_0 when ϕ and m is smaller. This is the opposite with the analysis in the previous paragraph. Eventually, we chose a balanced parameter shown in Table. I, which allows Rollbot to vary its revolving radius from 0.12 m to 1.28 m when $\omega_0 \in [0, 3\pi]$ and recover from transients within a few seconds.

E. Control Design

We already have a robot that can roll on the ground in circular pattern and theory derived in section II-B tells us

Quantity	Value	Explanation
R	0.12 m	Outer radius of shell
M	0.840 kg	Mass of shell
I	$0.0053\,\text{kg}\cdot\text{m}^2$	Moment of inertia of shell
ϕ	$45°$	Angle between the pendulum mass, center of shell and the axis of motor
r	0.093 m	Distance between pendulum mass's center of mass and center of shell
m	0.306 kg	Mass of pendulum mass
m_b	0.040 kg	Total mass of beads
k_0	$(0.4\,s^{-1})MR^2$	Friction coefficient

TABLE I
PHYSICAL PARAMETERS OF ROLLBOT.

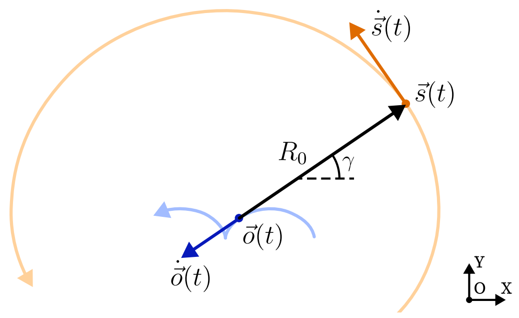

Fig. 7. Illustration of relevant parameters in the control algorithm. The orange and blue curves are the trajectory of the center of the shell and the corresponding center of curvature respectively. At time t, the center of shell is at $\vec{s}(t)$, the center of curvature is at $\vec{o}(t)$ and the curvature radius is R_0. The center of shell is moving at velocity $\dot{\vec{s}}(t)$ along the curve perpendicular to $\vec{s}-\vec{o}$, and if Rollbot is accelerating, then the center of curvature will also move along $\vec{s}-\vec{o}$ at a rate of $\dot{\vec{o}}(t)$

that the revolving radius R_0 can be modulated by changing the driving speed ω_0. In this section, we will derive a way of controlling Rollbot's 2D motion by changing R_0 at different times.

Instead of controlling the position $\vec{s}$ of Rollbot, we change to control the center of curvature $\vec{o}$.

$$\vec{o} = \vec{s} - R_0\{\cos\gamma, \sin\gamma, 0\} \tag{30}$$

When Rollbot is rolling with a constant driving speed ω_0, it will move in a circle, and the center of curvature will be stationary. However, if we accelerate the driving speed at a rate β at $t = t_0$ when $\vec{s}-\vec{o}$ is at angle γ, then the rate of change of curvature will be

$$|\dot{\vec{o}}| = v_{R_0} = \left.\frac{\mathrm{d}R_0(\omega)}{\mathrm{d}\omega}\right|_{\omega=\omega_0} \beta \tag{31}$$

and subsequently the velocity of center of curvature will be

$$\dot{\vec{o}} = -v_{R_0}\{\cos\gamma, \sin\gamma, 0\} \tag{32}$$

To move $\vec{o}$ towards a certain target position $\vec{o}_g$, a simple solution could be

$$v_R = -k_p(\vec{o}_g - \vec{o}) \cdot \{\cos\gamma, \sin\gamma, 0\} \tag{33}$$

where $k_p > 0$ is a constant coefficient. It is worth noting that in the experiments, $\vec{o}$ will not be the same as the actual center of curvature due to transient behavior of Rollbot.

In reality, we need to consider the capability of the motor and various non-idealities. So, instead of using (33), we add a limit to the revolving radius to limit the driving speed, a limit to the maximum acceleration so we are always close to the quasi-static state, and use PID instead of simple proportional control to compensate for external interference like the slope of the ground and to keep the system stable. The control diagram is shown in Fig. 8.

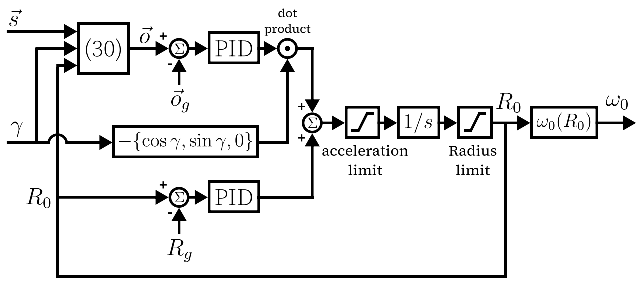

Fig. 8. Control diagram used in experiments. The input of the system are the current position $\vec{s}$ and orientation γ of the shell provided by the Optitrack system and the output is the target driving speed ω_0.

So far we have been talking about how to control $\vec{o}$ instead of $\vec{s}$. In case we want the robot to move across certain point $\vec{s}_g$ at velocity $\dot{\vec{s}}_g$, we only need to compute the corresponding target R_g using equations obtained in section II-B and target $\vec{o}_g$ using (30), and command Rollbot to roll around $\vec{o}_g$ at revolving radius R_g. If we want to stop the Rollbot at $\vec{s}_g$, we will first let it revolve around $\vec{o}_g$ and stop the motor when Rollbot is $\Delta t \approx 0.15\,\text{s}$ away from reaching $\vec{s}_g$, and the Rollbot will settle around $\vec{s}_g$.

III. Experiments

We conducted several experiments to demonstrate the ability of Rollbot. We used optitrack system to provide Rollbot with its real-time position and orientation, but all other computation are executed on-board.

A. Open-loop Motion

In the open-loop motion experiments, Rollbot will maintain a constant driving speed from 0 to 7 rad/s. We collect the revolving radius of the Rollbot under different driving speeds and compare it with the theoretical result obtained in section II-B to verify that our theory and approximation is consistent with reality. The results are shown in Fig. 9.

B. Stable Circular Motion

In the stable circular motion experiments, Rollbot will try to circle around a set point with a given radius ranging from 20 cm to 65 cm. Fig. 10 shows how Rollbot can start from a point far away from the setpoint and move towards the goal, and Fig. 11 shows the trajectory of Rollbot when it is stably revolving around the set point.

From Fig. 10 we can see that Rollbot move towards set point by having higher driving speed when moving towards

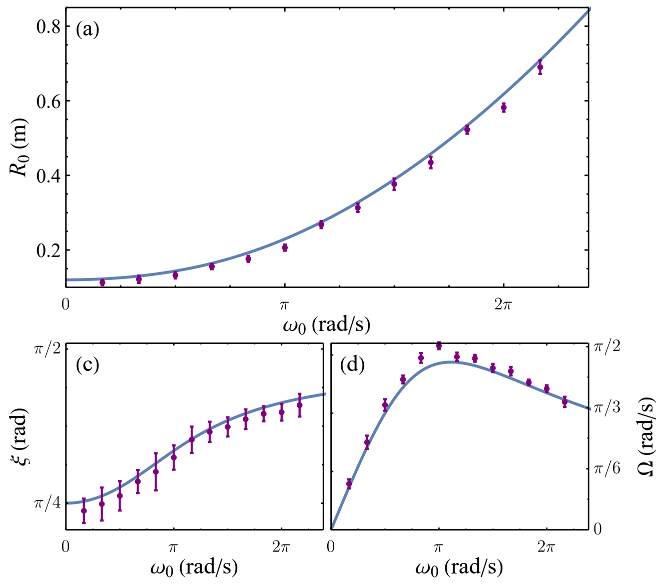

Fig. 9. Comparison between the theoretical and experimental results. Theoretical curves are drawn in blue lines and experimental results are drawn in purple points. The slight systematic difference between experimental data and theoretical prediction of R_0 and ξ (the difference is 6 mm and 1°respectively) are caused by the slight imbalance of the shell.

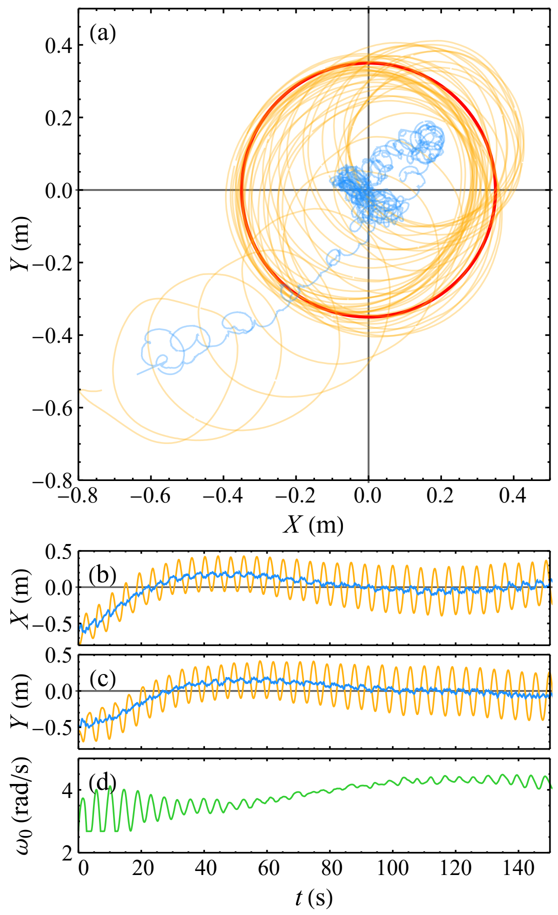

Fig. 10. Trajectory of Rollbot when moving towards the set point. (a) shows the trajectory, where the red circle shows the target radius $R_g = 0.35\,\text{m}$, and the orange and blue lines are the trajectory of Rollbot and the $\vec{o}$ respectively. (b), (c), and (d) shows the change of variables over time, where the orange and blue lines represent the position of Rollbot and the $\vec{o}$ respectively and the green line represent the driving speed.

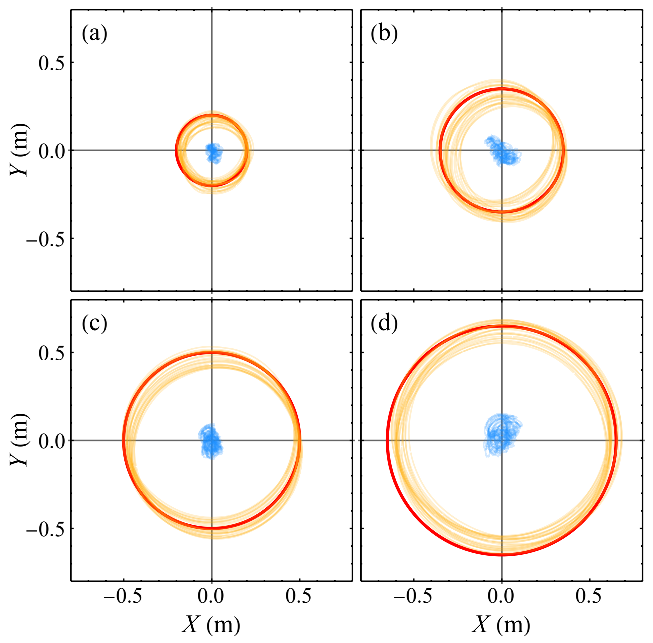

Fig. 11. Trajectory of Rollbot when stably revolving around the set point at $\{0{,}0\}$. Target revolving radius R_g is $0.20, 0.35, 0.50, 0.65\,$m for (a) to (d) respectively.

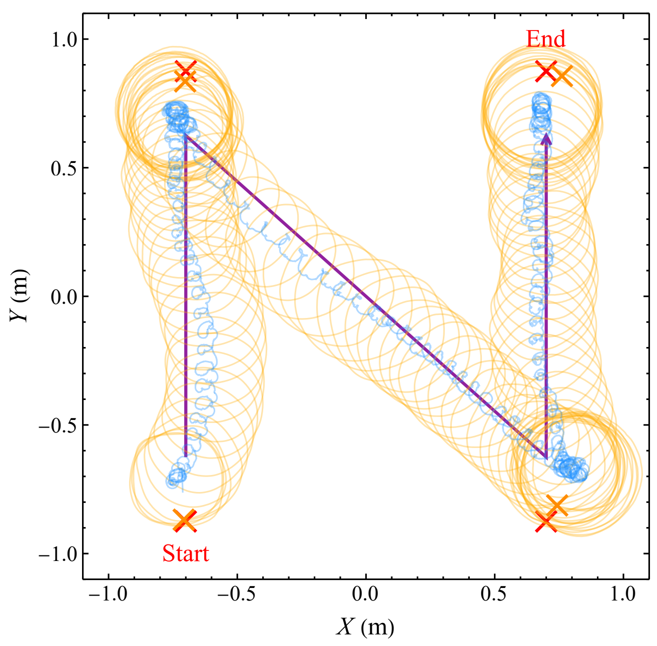

Fig. 12. Experimental results for waypoint movement experiment. Purple line shows the target trajectory of $\vec{o}$, red and orange crosses are the target and actual stopping points.

the set point and vice versa. In this example, Rollbot moves at an average speed of approximately 2 cm/s when moving towards the set point. Once it reached the stably revolving state, it will still modulate the driving speed slightly to compensate for the slope of the ground.

C. Moving between Waypoints

We would also like to demonstrate the Rollbot's ability to maneuver between waypoints, so we programmed Rollbot to move in a 'N' shape and stop at the four vertexes. The results in Fig. 12 shows that Rollbot could successfully move towards waypoints and stop within 7 cm of the targets.

IV. Conclusion

In this paper, we presented Rollbot, a single actuator spherical robot capable of controlled rolling motion on the ground. We provided a general mechanical analysis of spherical rolling robots driven by internal masses and the quasi-static state and stability analysis of Rollbot. Based on our theory, we designed Rollbot's hardware and control algorithm and successfully achieved stable circular motion and waypoint following. Through experiments, we also verified our theory's accuracy.

In future work, we plan to integrate an IMU on board to eliminate the need of using external Optitrack system, and we also plan to investigate more on how to design the control and path planning algorithm to make it move faster towards targets, have less steady state error, and potentially maneuver around obstacles or non-flat surfaces. With these future additions, Rollbot would be the simplest yet still powerful spherical robot, making it a great testbed for underactuated robotics and related fields.

References

[1] Bin He, Shuai Wang, and Yongjia Liu. "Underactuated Robotics: a Review". In: *International Journal of Advanced Robotic Systems* 16.4 (2019), p. 1729881419862164.

[2] Olfa Boubaker. "The Inverted Pendulum Benchmark in Nonlinear Control Theory: a Survey". In: *International Journal of Advanced Robotic Systems* 10.5 (2013), p. 233.

[3] Yanbin Feng Hongxing Li Jiayin Wang and Yundong Gu. "Hardware Implementation of the Quadruple Inverted Pendulum with Single Motor". In: *Progress in Natural Science* 14.9 (2004), pp. 822–827.

[4] Clement Gosselin, Frederic Pelletier, and Thierry Laliberte. "An Anthropomorphic Underactuated Robotic Hand with 15 DOFs and a Single Actuator". In: *ICRA*. 2008, pp. 749–754.

[5] Tadaaki Hasegawa et al. "Powerful and Dexterous Multi-finger Hand Using Dynamical Pulley Mechanism". In: *ICRA*. 2022, pp. 707–713.

[6] Wu Licheng, Kong Yanxuan, and Li Xiali. "A Fully Rotational Joint Underactuated Finger Mechanism and Its Kinematics Analysis". In: *International Journal of Advanced Robotic Systems* 13.5 (2016).

[7] Raffaele Di Gregorio. "Kinematic Analysis of the (nS)-2SPU Underactuated Parallel Wrist". In: *Journal of Mechanisms and Robotics* 4.3 (June 2012), p. 031006.

[8] David Zarrouk and Ronald S Fearing. "Compliance-based Dynamic Steering for Hexapods". In: *IROS*. IEEE. 2012, pp. 3093–3098.

[9] Evandro Bernardes, Frédéric Boyer, and Stéphane Viollet. "Modelling, Control and Simulation of a Sin-

gle Rotor UAV with Swashplateless Torque Modulation”. In: *Aerospace Science and Technology* (2023), p. 108433.

[10] Jianguo Zhao et al. “MSU Jumper: A Single-motor-actuated Miniature Steerable Jumping Robot”. In: *IEEE Transactions on Robotics* 29.3 (2013), pp. 602–614.

[11] James Kyle et al. “The Simplest Walking Robot: A Bipedal Robot with One Actuator and Two Rigid Bodies”. In: *2023 IEEE-RAS 22nd International Conference on Humanoid Robots (Humanoids)*. IEEE. 2023, pp. 1–7.

[12] Gedaliah Knizhnik and Mark Yim. “Thrust Direction Control of an Underactuated Oscillating Swimming Robot”. In: *IROS*. 2021, pp. 8665–8670.

[13] Andrew SaLoutos and Michael Rubenstein. “SpinBot: An Autonomous, Externally Actuated Robot for Swarm Applications”. In: *Distributed Autonomous Robotic Systems*. Springer, 2019, pp. 211–224.

[14] Jingxian Wang and Michael Rubenstein. “PCBot: a Minimalist Robot Designed for Swarm Applications”. In: *IROS*. IEEE. 2022, pp. 1463–1470.

[15] Satoshi Ito et al. “An Autonomous Mobile Robot with Passive Wheels Propelled by a Single Motor”. In: *Robotics and Autonomous Systems* 122 (2019), p. 103310.

[16] Weixuan Zhang, Mark W Mueller, and Raffaello D’Andrea. “A Controllable Flying Vehicle with a Single Moving Part”. In: *ICRA*. 2016, pp. 3275–3281.

[17] Andrew G Curtis et al. “Autonomous 3D Position Control for a Safe Single Motor Micro Aerial Vehicle”. In: *IEEE Robotics and Automation Letters* (2023).

[18] Aminata Diouf et al. “Spherical Rolling Robots - Design, Modeling, and Control: a Systematic Literature Review”. In: *Robotics and Autonomous Systems* (2024), p. 104657.

[19] Aarne Halme, Torsten Schonberg, and Yan Wang. “Motion Control of a Spherical Mobile Robot”. In: *Proceedings of 4th IEEE International Workshop on Advanced Motion Control*. Vol. 1. IEEE. 1996, pp. 259–264.

[20] Puyan Mojabi et al. “Introducing August: a Novel Strategy for an Omnidirectional Spherical Rolling Robot”. In: *ICRA*. Vol. 4. IEEE. 2002, pp. 3527–3533.

[21] Jiazhen Chen et al. “Design and Motion Control of a Spherical Robot with Control Moment Gyroscope”. In: *3rd International Conference on Systems and Informatics (ICSAI)*. IEEE. 2016, pp. 114–120.

[22] Shourov Bhattacharya and Sunil K Agrawal. “Spherical Rolling Robot: A Design and Motion Planning Studies”. In: *IEEE Transactions on Robotics and Automation* 16.6 (2000), pp. 835–839.

[23] Keith W Wait, Philip J Jackson, and Lanny S Smoot. “Self Locomotion of a Spherical Rolling Robot Using a Novel Deformable Pneumatic Method”. In: *ICRA*. IEEE. 2010, pp. 3757–3762.

[24] Vakhtang Putkaradze and Stuart Rogers. “On the Dynamics of a Rolling Ball Actuated by Internal Point Masses”. In: *Meccanica* 53 (2018), pp. 3839–3868.