Trajectory Optimization of Tail-Assisted Reorientation

Boan Tao, Chaojie Feng, Jingwen Zhang and Zhijie Wang

***Abstract*— This paper proposed a novel robotic system driven by a sequence of shoot and drag actions. One main sub-problem of the robotic system is addressed as a trajectory optimization problem, which can be transformed into a non-linear programming with Direct Collocation algorithms. A method to modify Direct Collocation algorithms to solve optimal trajectory as well as the initial state of the dynamical system is studied. The simulation results are evaluated in this paper while a prototype of tail-assisted block is built for future practical evaluations.**

I. INTRODUCTION

The robot community is always aiming to build a robust and versatile robotic system. In terms of robustness, the reaction of the robot to large external perturbations should be explored. This is a critically important feature for any dynamically-stable robot. As for versatility, we can imagine three regular scenarios: jumping, climbing ladders and walking through all kinds of terrains. It is hard to say that there is an off-the-shelf robot who can handle all of them perfectly. But many robotic applications always require these capabilities, such as the MARS exploration, disaster relief and surveillance.

Intuitively, humans perform these tasks perfectly with two

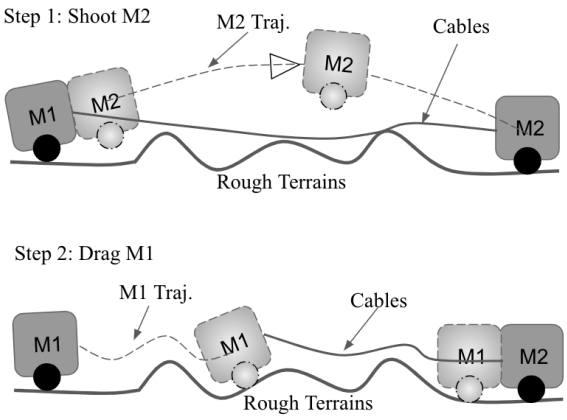

Fig. 1. An illustration of the Shooting-and-Drag Robotic System aiming to develop a low-cost and versatile robotic system. The system consists of two blocks and they can be docked together while they are also connected with cables. (Step1) $M2$ is shot out by $M1$ with an initial velocity and angle to reach desired position in a feasible pose, during which all kinds of rough terrains can be "jumped" over. (Step2) $M1$ is dragged back to a new docking state with $M2$, during which $M1$ can go over rough terrains adaptively. This paper focus on generating optimal $M2$ trajectory using direct collocation algorithms

legs. Legged robots have been good candidates to achieve this goal. Many complex planning algorithms and feedback control laws need to be developed while many limitations are still there. The cost of that kind of robots is very high and the solution lacks the generality when considering different kinds of terrains. For all kinds of terrains, wheeled robots are of good potentials, like the MARS rover. But jumping and climbing are not easy to achieve.

As illustrated in Fig. 1, we proposed a new robotic system which is consisted of two blocks $M1$ and $M2$. The two blocks are connected with cables.To drive this robotic system, the first step is to shoot $M2$ with an initial velocity and angle so that the block $M2$ can reach a desired position with correct orientation via a smooth trajectory. Then the second step is to drag the block $M1$ back with cables. Repeatedly, this robotic system can go over any kinds of terrains this simple formulation and achieves a good versatility with low cost.

Rather than attempt to develop the entire new robotic system, we focus on finding the optimal trajectory of block $M2$ in Step 1. We first formulate the problem as a traditional trajectory optimization, which turns out to find a trajectory for the dynamical system that satisfies some set of constraints while minimizing some cost functional [1]. Transcription methods for solving this optimal control problem by converting a continuous problem into a non-linear programming problem. Although Direct Collocation [2] is good candidate to get the smooth trajectory, we still need to find the optimal shooting velocity and angle that are part of initial states. We then present a method to modify Direct Collocation to generate optimal trajectory of $M2$, as well as part of initial states.

Our contributions include the following: *i*) A novel robotic system driven by a sequence of shoot and drag actions is proposed. This robotic system keep a good potential to achieve high versatility with low cost. *ii*) A method for modifying Direct Collocation algorithms into a non-linear programming which optimize the trajectory as well as the initial state of a dynamical system.

The remainder of this paper is organized as follows. In sec.II, we cover related work. In sec.III, we present the general formulation of our trajectory optimization problem. In sec.IV, we outline our both experimental simulation setup and hardware setup individually. In sec.V, we evaluate our methods. Finally in sec.VI and sec.VII, we provide future works and conclude.

II. RELATED WORK

Changing orientation during free fall has been an important topic in robotics system. In many cases, it can be considered as an inverted pendulum system and multiple

approaches have been conducted in this fields. Tails has been utilized by animals to store angular momentum to control body pose. By actuating a tail with a relatively high moment of inertia, rapid changes in gross body orientation are possible. The Tailbot used such mechanism to enable air-righting and traversing rough terrain [3]. Another approach is applying flywheels to provide angular momentum while enabling a smaller footprint and concrete design. ETH Zurich developed a system called Cubli that can jump up and balance on one corner. Actions can be made by suddenly stopped the momentum wheels and apply controlled motor torques [4]. Another approach is changing the shape of the body in the air to change the principle moment of inertia. Therefore the angular velocity of the body can be controlled. Disney Research used such principles to develop the Stickman, the robot that can perform gymnastics [5].

Optimal control methods are accessible to them which is divided to three main categories, dynamic programming, indirect and direct approaches. The core of dynamic programming is optimality of sub-arcs combining with recursive computation of feedback control for all time and state. But such method is constrained to small state dimension. Indirect methods optimize infinite problem basing on constraints which is then discretized to boundary value problem. The methods can be negatively influenced by nonlinearity and instability which leads to frequent update of control structure. As for direct methods, original infinite stats are discretized and transcribed to finite dimensional nonlinear programming problem (NLP) which can be solved by several effective numerical optimal solver. Conversely, direct multiple shooting method solves ODE on each interval $[t_i, t_{i+1}]$ to get state trajectory starting with artificial initial value. By combination of discretized controls and state sequence, NLP is found out constrained by path constraints, terminal constraints, initial value and continuity. Because state sequences is kept as optimization variables within the NLP, this method can control unstable and nonlinear systems perfectly with initialization of state and other constraints [6].

III. PROBLEM STATEMENT

The aim of this project is implementing and improving direct multiple shooting algorithm to optimize shooting trajectory while reaching goal location with desired orientation. The cost function of such nonlinear programming problem is designed as implicit function of motor torque to maximize energy efficiency. Additional physical constraints are also considered, like states range in trajectory and control input limitation which is the torque limitation of the motor controlling the rotation of the tail. The kernel of this method is applying cubic polynomial to represent the approximated trajectory on given segment and then minimizing the defect at center of the segment between state derivatives from dynamics and from cubic approximation. The corresponding schematic diagram is shown in Fig. 2. To obtain the expression of defect Δ_k, subinterval from $[t_k, t_{k+1}]$ is chosen with two nodes X_k and X_{k+1}.

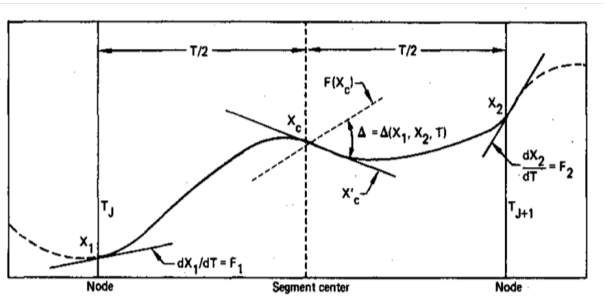

Fig. 2. Direct multiple shooting method schematic diagram [2].

$$\begin{aligned} X(0) &= X_0 \\ X(h) &= X_h \\ \dot{X}(0) &= \dot{X}_k = f(X_k, u_k) \\ \dot{X}(h) &= \dot{X_{k+1}} = f(X_{k+1}, u_{k+1}) \end{aligned} \tag{1}$$

The interval is then shifted to $[0,h]$ to get statement equations set. The solution of it gives unique coefficients of the cubic polynomial which can help deducing time-derivative at midpoint $\dot{X}_c$. The defect Δ_k is constructed by the difference between $\dot{X}_c$ and state derivative at same point from the system dynamics $f(x,u)$.

$$\dot{X}_c = -\frac{3}{2h}(X_k - X_{k+1}) - \frac{1}{4}[f(X_k, u_k) + f(X_{k+1}, u_{k+1})] \tag{2}$$

$$\Delta_k = (X_k - X_{k+1}) + \frac{h}{6}[f_k + 4f_c + f_{k+1}] \tag{3}$$

Therefore, the optimal problem can be described in the NLP form. To improve the algorithm in this project, initial state assignment is replaced by suitable bounds to find optimal initial velocity and its angle.

$$\begin{aligned} \underset{x_1,\ldots,x_N,u_0,\ldots,u_{N-1}}{\text{minimize}} \quad & \sum_{n=0}^{N-1} g(x_n, u_n)dt \\ \text{subject to} \quad & \Delta_k = 0 \quad \forall k \in [1,N], \\ & X(0) = X_0, \\ & X(t_f) = X_f, \\ & u_{min} \le u_k \le u_{max}, \\ & X_{min} \le X_k \le X_{max} \end{aligned} \tag{4}$$

IV. EXPERIMENTAL SETUP

A. Dynamics

The dynamics of the problem is simplified into a tail-assisted block with initial shooting velocity v_0 and angle θ_{b0} assuming that it is constrained as a two-dimensional problem and air drag is negligible. As illustrated in Fig. 4, the trajectory of the block is not solely a parabola due to the coupling of tail rotation. Assuming the rotation center of the tail coincide with the center of the block, the acceleration of COM of the block will be influenced by the rotation of the tail.

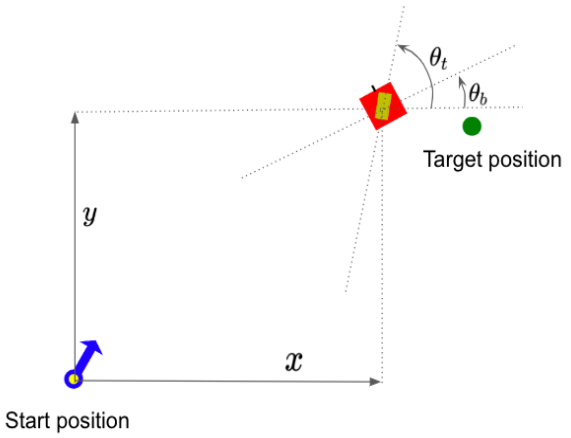

Fig. 3. An illustration of simulation setup which describes the dynamics of the system. The red block represents the main body of the block $M2$ while the yellow rectangle represents the tail which is used to correct the pitch angle of the main body. The blue arrow indicates the initial shooting velocity and angle.

Given the motor torque τ, tangential acceleration due to the rotation of the tail is expressed as follows:

$$a_t = \frac{\tau}{\rho m_b} \tag{5}$$

Where ρ is the off-center distance between the COM of the block and the point mass of the tail and m_b is the mass of the block. The COM translational accelerations can be obtained as:

$$\begin{cases} \ddot{x} = \frac{\tau}{\rho m_b}\sin(\theta_b + \theta_t) \\ \ddot{y} = -g - \frac{\tau}{\rho m_b}\cos(\theta_b + \theta_t) \end{cases} \tag{6}$$

Where θ_b and θ_t are the pitch angles of the block and the tail. As for the angular accelerations, total angular momentum as follows is kept without external disturbance [7].

$$\begin{aligned} H_0 &= I_b\dot{\theta}_b + I_t\dot{\theta}_t \\ I_t &= \frac{m_b m_t}{m_b + m_t}\rho^2 \end{aligned} \tag{7}$$

Where I_b is the mass moment of inertia of the block while I_t is the equivalent moment of inertia from the point mass of the tail. Assuming that the initial shooting velocity always located at the COM of the block which will not generate any initial angular momentum, $H_0 = 0$. The relationship between angular accelerations can be obtained as:

$$\begin{cases} \ddot{\theta}_b = \frac{1}{I_b}\tau \\ \ddot{\theta}_t = -\frac{1}{I_t}\tau \end{cases} \tag{8}$$

The dynamics of the system can then be written using state-space formulation as:

$$\begin{aligned} \dot{X} &= f(X,u) \\ X &= \begin{bmatrix} x & y & \theta_b & \theta_t & \dot{x} & \dot{y} & \dot{\theta}_b & \dot{\theta}_t \end{bmatrix}^T \end{aligned} \tag{9}$$

Where the motor torque τ is the only input as u. With this dynamics, we can generate a feasible trajectory with simple controllers, like Bang-Bang control, as the initial guess of our optimization problem. At the same time, the optimizer will give us the trajectory of X, as well as the control sequence u. We can use dynamics with this control sequence u to simulate the state trajectory, which is used to validate the result of optimization. Meanwhile, the shooting velocity and angle show up in the initial state as follows:

$$\begin{cases} \dot{x}_0 = v_0\cos\theta_{b0} \\ \dot{y}_0 = v_0\sin\theta_{b0} \end{cases} \tag{10}$$

Following section will present our method to solve the trajectory and the initial state simultaneously based on Direct Collocation algorithms.

B. Simulation Setup

Traditional direct collocation method proposed by C.R.Hargraves and S.W.Paris [2] is able to solve for an optimized control sequence with known initial state. Advanced Booster is an example of maximizing final weight that can be injected by rocket into the earth's orbit with optimal thrust and known initial state. The ultimate goal in this paper aims to find a feasible initial state with a sequence of minimal control input to reach the final state. Modifications were made to the traditional approach with the use of system dynamics to address the problem. The system dynamics have two underlying purposes: 1) To generate a feasible trajectory between initial state and final state. 2) To validate the states trajectory from direct collocation. Sequential Quadratic Programming is used to solve this nonlinear optimization problem.

1) Initial State Consideration: If the initial position $[x_0, y_0]$ of the block is unknown, the problem will be intractable because the search space for eight states is too large. So the problem limits to a known initial position. Even if initial position is fixed, search space is still large and solutions are not guaranteed. In order to find such initial states, firstly, assume that initial states for the block is given as $X_{bi} = [x_0, y_0, \theta_{b0}, \dot{x}_0, \dot{y}_0, \dot{\theta}_{b0}]$, and the only constraint for the block's final state is θ_{bf}. Based on this assumption, there must exist a feasible trajectory by using dynamics and simple controller (i.e. PID, LQR, etc). Thus, all of the block's final states $X_{bf} = [x_f, y_f, \theta_{bf}, \dot{x}_f, \dot{y}_f, \dot{\theta}_{bf}]$ can be observed. Obviously, number of solutions can only possibly increase if final states from X_{bf} are partially constrained based on user requirements. Then by loosing the initial condition X_{bi} with bounds, an optimal trajectory $X = [X_1, X_2, ...X_N]$ can be generated along with control input $u = [u_1, u_2, ..., u_N]$. The optimal trajectory can be verified by using system dynamics with input u.

2) Time Consideration: Initial state consideration can only prove the existence and validity of solution given by direct collocation with final states constrained because it assumes an initial state is given to generate system dynamics. In practice, initial condition is not known, neither can be assumed. Also, time T is another implicit constraint which is highly related to trajectory. In fact, different trajectories between initial and final states represent time difference. As a result, T can be bounded in $(0, T_{max})$ and added to decision variables as $[u_1, u_2, ..., u_N, X_1, X_2, ...X_N, T]$ to increase the

flexibility of solution. By this approach, a feasible states trajectory with initial condition and best control strategies can be found directly.

C. Hardware Setup

To explore the effects of orientation control in an aerial maneuver, we built a robot prototype with an active tail to control the orientation in one degree of freedom. The prototype consists of a 3D printed body, a Raspberry Pi 3B+ to provide integrated control, a MPU9250 IMU, a reactive wheel and a FSR90 servo motor.

The robot body is first modeled in CAD so that we can easily find the center of mass (COM) of the system. The motor is acting at COM so it wont generate angular moment in other axis. The body is designed with minimal weight because of motors limitation. The reactive wheel is 3D printed with ABS material. A M4 nut-screw is attached to the wheel acting as a point mass. The servo motor has a max rotation speed of 130 RPM, max torque of 1.5 kg-cm and only needs 5V servo signals to drive, external motor controlled is not required which simplifies the design.

Here we applied 9 degree of freedom IMU consists of gyroscope, magnetometer and accelerometer. Because the accelerometer data is only reliable on the long term and gyroscopes data is only reliable on the short term we used a complementary filter to interpret IMU data to angular position. The gyroscope data is integrated every timestep wit the current angle value, after it is combined with the low-pass data from the accelerometer. The PID controlled and IMU interface is programmed in python in a Raspberry pi 3B+, which is controlled remotely through wifi.

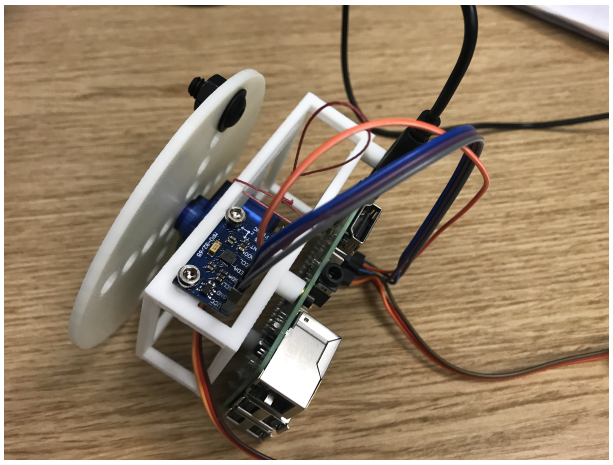

Fig. 4. An illustration of hardware setup

V. Experimental Results

A. Simulation

1) Verification: Based on simulation set-up, one-second perform analysis is conducted to validate whether methods described in *Initial State Consideration* is achievable. Assume the block is shooting at 40 m/s with an angle at 60 degrees. The initial state becomes $X_i = [x_0 = 0, y_0 = 0, \theta_{b0} = 60^o, \theta_{t0} = 0^o, \dot{x}_0 = 40cos(60^o), \dot{y}_0 = 40sin(60^o), \dot{\theta}_{b0} = 0, \dot{\theta}_{t0} = 0]$. After one second, the block's final state is observed as X_f. By running direct collocation, a sequence of best control strategies u and state trajectories X are generated. A comparison between state trajectories using dynamics with input u and directly from collocation can be plotted.Note that with N = 50 points, there's almost no error between each other, which suggests the model is correct (shown in Fig. 5 and Fig.6).

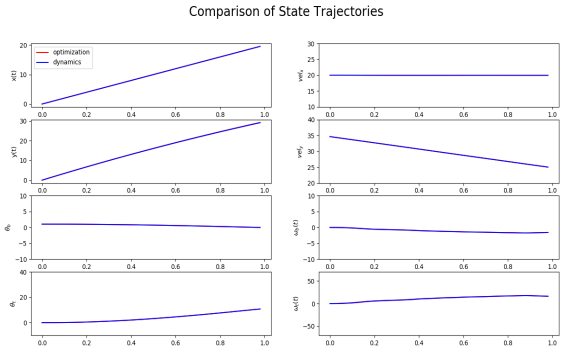

Fig. 5. Comparison between state trajectories. Red curve uses dynamics model with input sequence from direct collocation. Blue curve uses states directly from direct collocation. N = 50

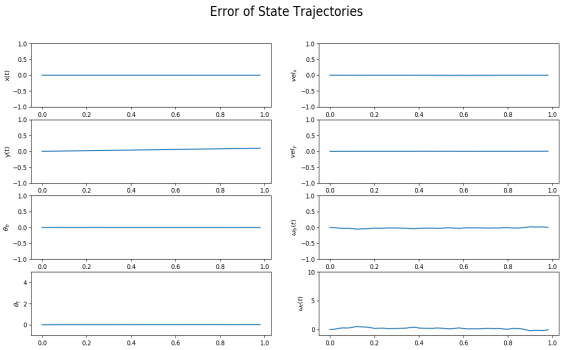

Fig. 6. Differences in state trajectories between using dynamics with input torque and directly using direct collocation states. N = 50

2) Initial State Consideration: After verifying the model, the block's shooting velocity and shooting angle are put as unknown variable, and based on observed final state, $X_f = [x_f = 19.55, y_f = 29.25, \theta_{bf} = 0]$ is added into constraint. Re-run direct collocation algorithm based on modified constraint with unknown initial shooting velocity and angle. The one-second performance plots are shown in Fig. 7 and Fig. 8. Both shooting velocity and shooting angle are calculated as $\theta_{bi} = 59.99^o$ and $v_i = 40.03m/s$. This result suggest an inverse approach to derive initial state is possible.

3) Time Consideration: A further modification is made on time, which will include time-to-destination T as a decision variable to improve the flexibility and robustness of model to solve more practical problems. Assume the final state is partially constrained with block's target position $[100, 0]$ and

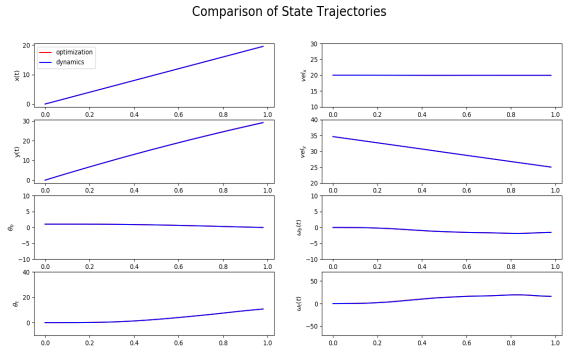

Fig. 7. Comparison between state trajectories without known initial condition. Red curve uses dynamics model with input sequence from direct collocation. Blue curve uses states directly from direct collocation. N = 50

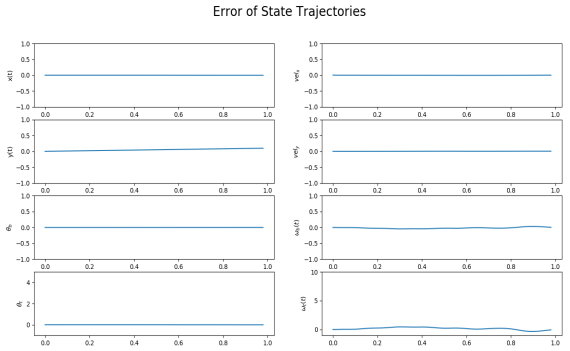

Fig. 8. Differences in state trajectories between using dynamics with input torque and directly using direct collocation states. The initial condition is unknown. N = 50

orientation as 0^o, and initial shooting velocity v_i is bounded as $(0,80)$. Time T is bounded in $(0,10)s$. By running direct collocation with time consideration, state trajectories are plotted in Fig. 9 and Fig. 10 with error between system dynamics and direct collocation. Note that there are slightly difference in positional state as time increases. This occurs because the dynamic modeling uses 1st order approximation, and number of collocation points N is not big enough so that small discrepencies are being accumulated. The motor maintains an efficient way to control to block's orientation as expected. Eventually, the initial state of block can be calculated. Shooting velocity is $v_i = 51.27m/s$ and shooting angle is $\theta_{bi} = 10^o$.

B. Hardware

Here we tested prototypes ability to achieve orientation control. The robot is suspand with a string on its top edge and initialized at zero degree in roll axis. Then we give the robot a slight push so that it starts to swing around roll axis. When control is not implemented, the robot swings back and forth for more than 5 seconds. When PID control is implemented, as soon as we pushed robot, the wheel starts to rotate to counterbalance the torque. And the robot converges to stabilization in less than 2 seconds. The readings from

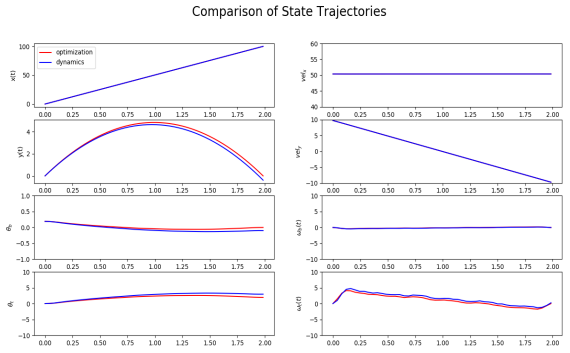

Fig. 9. A practical example. Comparison between state trajectories without known initial condition. Red curve uses dynamics model with input sequence from direct collocation. Blue curve uses states directly from direct collocation. N = 50

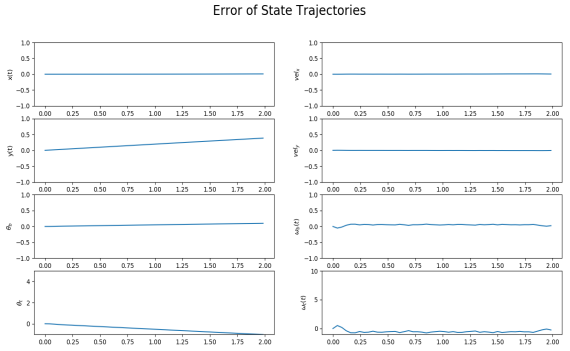

Fig. 10. A practical example. Differences in state trajectories between using dynamics with input torque and directly using direct collocation states. The initial condition is unknown. N = 50

IMU is shown in Fig. 11. We can come to a conclusion that the eccentric wheel can adjusting the bodys inertia to keep stability.

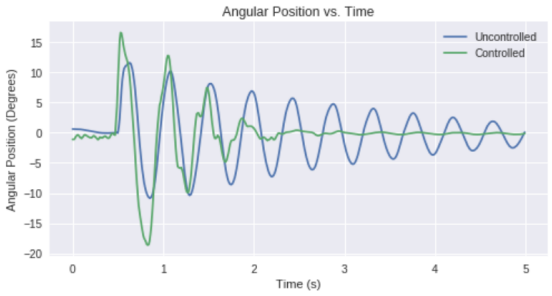

Fig. 11. Hardware Implementation Result

VI. FUTURE WORKS

There are several improvement to implement in future work. First, nonlinear programming problem need to be solved in each sampling segment and the algorithm keeps approximation and optimization alternately in a high frequency, which leads to high computational load. In the project,

we used Scipy to implement the Sequential Quadratic Programming (SQP) solver for NLP. Therefore, once the number of nodes increased, the running speed increased dramatically. Applying commercial solver software, like IPOPT, instead is an ameliorative method. Second, cost function in current state is $u^t Ru$ to achieve less control effort and maximum torque input efficiency. Other cost function such like $minimize(t_f)$ can be constructed to find the trajectory of the minimum time. Moreover, we can combine them together for considering both the time and control effort with cost function $minimize(\sum_{t=0}^{t_f} \gamma^t u^T Ru)$. Third, it is indispensable to acquire nodes state for calculating defect. Some real-time camera can be applied to record and process states information at each node. Fourth, in current stage, initial state is presented directly by value. Later scheme will transfer it to input value of shooting mechanism like torsional spring and flywheel.

VII. CONCLUSION

In this paper, we present a method to modify traditional Direct Collocation algorithms. Using this method, we can solve the optimal trajectory of some minimal cost, as well as the initial state, like the shooting velocity and angle in our dynamical system of the proposed robotic system. This allows us to find the optimal motor torque sequence controlling the rotation of the tail on block $M2$ and initial shooting states simultaneously. In simulation, we demonstrated that our method works very well and when given the goal position and pose of our shooting task, the optimizer gave us feasible and smooth trajectory for block $M2$ to reach it from the initial position, which is a core capability for the proposed robotic system.

After integrating this control algorithm into the entire robotic system in the future, the proposed novel robotic system can not only be used as an individual module to do exploration and surveillance, it can also work as a functional module to extend other robots' capabilities. Between two points where conventional robots are hard to find a way to move from one to another, this kind of robotic system can build a "tunnel" for them where other robots can use the cable to slide from current position to the goal. The concept of "robot's tools" can be developed with it to extend current robot's capabilities.

REFERENCES

[1] Matthew P. Kelly, "Transcription Methods for Trajectory Optimization: A beginners tutorial", *Tutorial, Cornell University*, July, 2017.

[2] C. R. Hargraves and S. W. Paris, "Direct trajectory optimization using nonlinear programming and collocation", *J Guidance, 10(4):338-342*, July-August, 1987.

[3] Zhao, Jianguo, et al. "MSU tailbot: Controlling aerial maneuver of a miniature-tailed jumping robot", *IEEE/ASME Transactions on Mechatronics*, 20.6 (2015): 2903-2914.

[4] Gajamohan, Mohanarajah, et al. "The cubli: A cube that can jump up and balance", *Intelligent Robots and Systems, 2012 IEEE/RSJ International Conference on. IEEE*, 2012.

[5] Pope, Morgan T., et al. "Stickman: Towards a Human Scale Acrobatic Robot", *2018 IEEE International Conference on Robotics and Automation (ICRA). IEEE*, 2018

[6] Moritz Diehl, Hans Georg Bock, Holger Diedam, Pierre-Brice Wieber. "Fast Direct Multiple Shooting Algorithms for Optimal Robot Control", *Fast Motions in Biomechanics and Robotics*, Heidelberg, Germany. 2005

[7] Aaron M. Johnson, E. Chang-Siu, T. Libby, Masayoshi Tomizuka, Robert J. Full, and D. E. Koditschek. "Tail assisted dynamic self righting", *in Proc. Intl. Conf. on Climbing and Walking Robots*, 2012.