Project: The Simplest Walking Model's Simulation

Full Name, Matrikelnummer: Dexiao Zhou, 23230864

Friedrich-Alexander-Universität Erlangen-Nürnberg
Machine Learning and Data Analytics Lab
https://www.mad.tf.fau.de/
Carl-Thiersch-Straße 2b, 91052 Erlangen, Deutschland
Email: dexiao.zhou@fau.de

***Abstract*—This report presents a detailed simulation and analysis of a model of an inverted pendulum walking on a slope. This model abstracts the complex of human gait into a bipedal system of point masses and rigid rods. The model allows insight into the system dynamics and stability characteristics of walking. To analyse the behaviour of the model under different initial conditions, we employ the basin of attraction method. This analysis provided insights into the correlation between initial parameters and gait stability. Among them, when applying the basin of attraction method for analysis, the influence of the slope angle on the stability of the model is mainly discussed, and it is revealed that the gait stability increases as the slope angle expands within a specific range. This project provides valuable insight into the further understanding and design of passive dynamic walking robots.**

I. Introduction

This report presents a simulation of a walking inverted pendulum along a slope, which simplifies the motion of human walking into a model consisting of two rods and three point masses. In this model, the mass of the human body (Point B) is much larger than the mass of the feet (Point A and Point C), while the mass of the leg (rods) is neglected. Refer to Figure 1 for illustration.

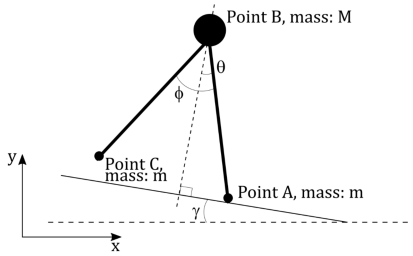

Fig. 1. Overview of the inverted pendulum model

In this simulation, the following parameters are used:

- Angle γ = 0.006 radian represents the angle of the slope.
- Length L = 12 represents the length of the simplified legs.
- Mass $m = 1 \times 10^{-10}$ represents the mass of the simplified point masses for Point A and Point C.
- Mass M = 1 represents the mass of the simplified point mass for Point B, which represents the body.

In this model, Point B represents the body simplified as a point mass with a mass of M. Point A and Point C represent the simplified point masses for the legs, both with a mass of m. The angle ϕ represents the angle between the two legs, and θ represents the angle between the front leg and the perpendicular line from Point B to the slope. The angle γ represents the angle of the slope.

Delving into the dynamics of the simplest walking models contributes to a deeper understanding of the mechanics of walking. In the context of walking robots and exoskeleton devices, the scope should extend beyond mere active control to a more profound comprehension of the intrinsic patterns of human gait. This serves to ascertain which parameters have a significant impact on stability and which parameter adjustments can enhance walking efficiency. It is noteworthy that during locomotion, individuals can direct their attention toward contemplation without overly focusing on the act of walking itself.[1] This ability allows for the simplification of intricate motion processes into fundamental models, enabling the exploration of the underlying principles of walking.

II. System Dynamics Model for the Swing Phase

Firstly, we model the dynamics of the bipedal inverted pendulum using the Lagrangian equations. We begin by calculating the kinetic and potential energies of each component. Then, we derive the Lagrangian operator by taking partial derivatives with respect to the generalised coordinates. Finally, we substitute the derived expressions into the Lagrangian equations to obtain the final dynamic equations. The process is as follows:

A. Kinetic and potential energies

First, we calculate the kinetic and potential energies of Point B. Since θ and ϕ are functions of time t, we express them as $\theta(t)$ and $\phi(t)$.

The kinetic energy T_M of Point B is:

$$T_M = \frac{1}{2} M L^2 \left(\frac{d\theta(t)}{dt} \right)^2 \tag{2.1.1}$$

The potential energy P_M of Point B is:

$$P_M = MgL\cos(\gamma - \theta(t)) \tag{2.1.2}$$

Then we calculate the potential energy of Points A and C. Since the masses of the two legs are equal, both being m, and

assuming the left leg is at the origin in the single support state, the kinetic and potential energy of the left leg are both zero. Therefore, the kinetic energy of the right leg T_{mr}, is given by:

$$\begin{aligned} T_{mr} = \frac{1}{2}mL^2 \Bigg(&\cos(\gamma-\theta(t))\frac{d\theta(t)}{dt}+ \\ &\cos(\gamma+\phi(t)-\theta(t))\left(\frac{d\phi(t)}{dt}-\frac{d\theta(t)}{dt}\right)\Bigg)^2 \\ &+\frac{1}{2}mL^2\left(\sin(\gamma+\phi(t)-\theta(t))\left(\frac{d\phi(t)}{dt}-\frac{d\theta(t)}{dt}\right)+\right. \\ &\left.\sin(\gamma-\theta(t))\frac{d\theta(t)}{dt}\right)^2 \end{aligned} \tag{2.1.3}$$

The potential energy of the right foot P_{mr} is:

$$P_{mr} = -mgL(\cos(\gamma+\phi(t)-\theta(t))-\cos(\gamma-\theta(t))) \tag{2.1.4}$$

By adding equations (2.1.1) and (2.1.3), we can obtain the total kinetic energy T_{all}. Similarly, by adding equations (2.1.2) and (2.1.4), we can obtain the total potential energy P_{all}. The Lagrangian operator is then obtained by subtracting the total potential energy from the total kinetic energy.

$$L_{\text{all}} = T_{\text{all}} - P_{\text{all}} \tag{2.1.5}$$

B. Differential Equations of dynamical system

By taking partial derivatives of the Lagrangian operator L_{all} (2.1.5) with respect to the generalised coordinates and substituting them into the Lagrangian equation

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) - \frac{\partial L}{\partial q_i} = 0$$

, the dynamic differential equations of the system can be obtained. To directly obtain the differential equations of the dynamical system, the function 'functionalDerivative' from Matlab can be used.

The differential equations of the system are as follows:

$$\begin{aligned} &M[2m-2m\cos(\phi(t))]L\ddot{\theta}(t)+[\cos(\phi(t))-1]mL\ddot{\phi}(t)+ \\ &mL\sin(\phi(t))[2\dot{\theta}(t)\dot{\phi}(t)-\dot{\phi}(t)^2]+ \\ &[(M+m)\sin(\gamma-\theta(t))-m\sin(\gamma+\phi(t)-\theta(t))]g=0 \end{aligned} \tag{2.2.1}$$

$$\begin{aligned} &L(1-\cos(\phi(t)))\ddot{\theta}(t)-L\ddot{\phi}(t) \\ &\quad+L\sin(\phi(t))\dot{\theta}(t)^2-g\sin(\gamma+\phi(t)-\theta(t))=0 \end{aligned} \tag{2.2.2}$$

With the mass matrix M_{mat}, the Coriolis matrix C_{mat}, and the gravity matrix G_{mat} are as follows:

$$M_{\text{mat}} = \begin{bmatrix} [M+2m-2m\cos(\phi(t))]L & [\cos(\phi(t))-1]mL \\ L(1-\cos(\phi(t))) & -L \end{bmatrix} \tag{2.2.3}$$

$$C_{\text{mat}} = \begin{bmatrix} 2mL\sin(\phi(t))\dot{\phi}(t) & -mL\sin(\phi(t))\dot{\phi}(t) \\ L\sin(\phi(t))\dot{\theta}(t) & 0 \end{bmatrix} \tag{2.2.4}$$

$$G_{\text{mat}} = \begin{bmatrix} [(M+m)\sin(\gamma-\theta(t))-m\sin(\gamma+\phi(t)-\theta(t))]g \\ g\sin(\gamma+\phi(t)-\theta(t)) \end{bmatrix} \tag{2.2.5}$$

III. The impact map of the Walking Inverted Pendulum

Now we analyse the relationship of the angles and the angular velocities between the situation "Before heel-strike collision" and the situation "After heel-strike collision", to obtain the impact map of the walking inverted pendulum.

And we assume:

1) An impact only occurs at point "c", and not at point "a". The model under this situation refers to a scenario where the walker's foot collides with the ground at a specific point (possibly the heel, as a "heel strike" is mentioned) without considering the possibility of other parts of the foot or body striking at the same time.
2) The impulsive force at the moment of impact is much greater than non-impulsive forces, such as gravity. This is a typical assumption in collision analysis where we assume the impulsive force is so great that it overwhelms all other forces at the moment of impact.

From Figure 2 and Figure 3, we can obtain the relationship of the angles as follows[3]:

$$\begin{aligned} \theta^+ &= \theta^- - \phi^- \\ \theta^+ &= -\theta^- \\ \phi^+ - \theta^+ &= -\theta^- \\ \phi^+ &= -\theta^- + \theta^+ \\ \phi^- &= 2\theta^- \\ \phi^+ &= -2\theta^- \end{aligned} \tag{3.1}$$

By using the Law of Conservation of Angular Momentum we can calculate the relationship between the velocities before heel-strike collision $\dot{\theta}^-$ $\dot{\phi}^-$ and the velocities after heel-strike collision $\dot{\theta}^+$ $\dot{\phi}^+$.

Then we obtain the final impact map as follows [2]:

$$\begin{bmatrix} \theta \\ \dot{\theta} \\ \phi \\ \dot{\phi} \end{bmatrix}^+ = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & \cos(2\theta^-) & 0 & 0 \\ -2 & 0 & 0 & 0 \\ 0 & \cos 2\theta^-(1-\cos 2\theta^-) & 0 & 0 \end{bmatrix} \begin{bmatrix} \theta \\ \dot{\theta} \\ \phi \\ \dot{\phi} \end{bmatrix}^- \tag{3.2}$$

IV. Dynamic simulation

The walking inverted pendulum is simulated through Matlab. The main steps are as follows:

- Define the parameters of the system.

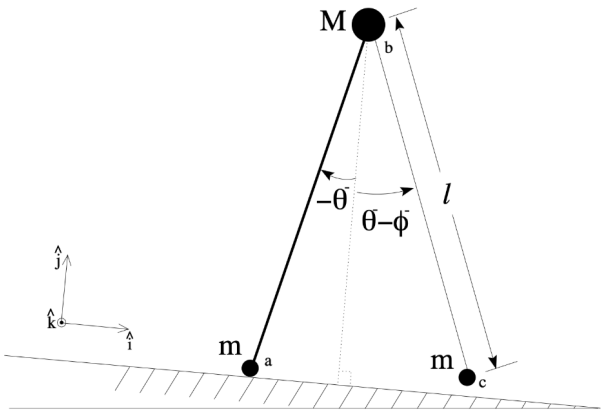

Fig. 2. Before heel-strike collision[2]

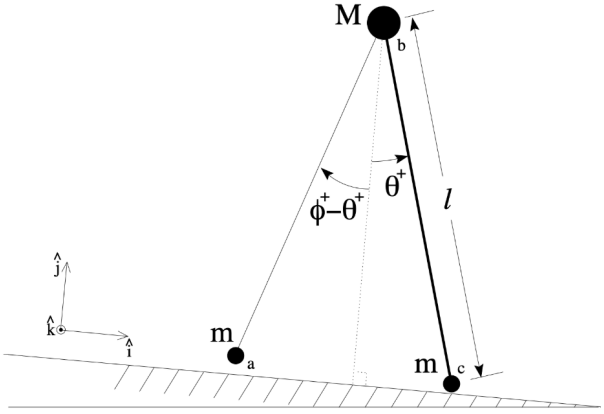

Fig. 3. After heel-strike collision[2]

- Assign the initial values.
- Designate the 'event function' for ode45 to handle events such as 'corrosion'.
- Create the main loop, and use ode45 to calculate the output values.
- Create the ODE(ordinary differential equation) function to compute the derivatives of the state variables.

A. The parameters of the system

The parameters of the system are as follows:

TABLE I
SIMULATION PARAMETERS

	γ	$B = m/M$	$steps$	per
Values	0.006	1×10^{-10}	10	5

which 'γ' is the angle of the slope, 'B' represents the ratio of the mass M at the Point B and the m at the Point A and C, 'steps' is the number of the steps the walking inverted pendulum has taken, and 'per' is the maximum time in seconds allowed for each steps.

Initially, we can choose 15 as the value for the 'per' parameter. After running the entire simulation, we find that each cycle takes about 3.86 seconds. To avoid any collisions that might occur beyond the calculation time of each step, we select 5 as the maximum time in seconds allowed for each step. This duration is sufficient for the simulation of each step.

B. The initial condition

The challenge of determining the initial value consistently arises when solving ODEs. In the second part of our approach, we derive the system's differential equations. However, without the initial conditions, it's impossible to solve the ODEs.

Therefore, we apply the 'Basin of Attraction' methodology to identify these initial conditions. Further elaboration on this technique will follow in the subsequent discussion.

C. Events function

Then a function for the collision of the inverted pendulum should be designed. This function will be the event function of the ode45.

The name of the function is collision, and it takes t and y as input parameters. t represents time, while y is the state vector of the system.

The main content within the function body checks for a specific collision condition. A collision is deemed to have occurred when $\phi - 2\theta$ equals 0. This condition is used to detect heel strike during the simulation of walking dynamics.

The function returns three values:

value: Represents the value of the collision condition. If this value is 0, it means the collision condition is satisfied.

isterminal: This flag informs the simulator whether it needs to stop the simulation when a collision occurs. In this case, it is set to 1, indicating that the simulation stops when a collision occurs.

direction: This parameter specifies the direction of the event. Here, its value is 1, which means that this function triggers an event only when value transitions from a negative value to a positive one, i.e., a collision is only considered to have occurred when $\phi - 2\theta$ increases from a negative value to zero.

D. Creation of the integral function

This is one of the most important functions, which is based on dynamical equations. We can use this function to compute the derivative of the state variables, obtaining the accelerations of both θ and ϕ. This function constitutes the main part of the ODE.

The calculation process is as follows: First, we set the constant parameters, which are the length of the leg L and the gravitational acceleration g. Then, we use the equations of the mass matrix M_{mat} (2.2.3), the Coriolis matrix C_{mat} (2.2.4), and the gravity matrix G_{mat} (2.2.5) to compute the acceleration.

Next, we compute the inverse of matrix M and calculate the equation:

$$eqn = M_{\text{mat}}^{-1}(-C_{\text{mat}} - G_{\text{mat}})$$

Note: To simplify the calculation, the matrix 'C_{mat}' is the product of the Coriolis matrix C_{mat} and the derivatives of θ and ϕ.

Then we obtain the derivative of the state variable:

$$\dot{y} = [\dot{\theta}\, \ddot{\theta}\, \dot{\phi}\, \ddot{\phi}]^T$$

E. Main Loop function

In the main loop function, there are several crucial aspects:

1) Setting the ODE options, which include establishing tolerances and defining the condition for collisions.
2) Refreshing the initial conditions after each collision.
3) Filtering results for normal walking.

Discussing the first point, to achieve a more accurate outcome, we set 'RelTol' to '1e-4' and 'AbsTol' to '1e-8'. We also assign the collision function as the ODE's condition.

Addressing the second point, after every collision, we extract the state variable's value from the last step. Utilizing the equations from the impact map (3.2), we can calculate the initial value for the subsequent step.

Regarding the third point, we need to determine whether the model is walking 'normally' or 'almost normally' at a given time. For instance, consider a simulation involving 5 steps. If the robot completes all 5 steps, we categorize it as 'normal'. If it accomplishes 3 or 4 steps, we classify it as 'almost normal'. However, if it can only manage one or two steps, we label it as 'faulty'.

If ϕ equals twice θ after one step, we consider the step as completed. However, this may not necessarily indicate walking, so we need to incorporate an additional evaluation method. For instance, both ϕ and θ should be less than 1 radian each time. We can also calculate the average of all θ values during one simulation, which must also be less than 1 radian. Another condition is that the simulation time should be longer than three times the steps. This is used to assess whether the steps are too short.

Moreover, to ensure that each step proceeds normally, I added a conditional check to determine whether the θ value at each step is less than 1. The aim of this step is to filter out any incorrect simulations, ensuring that the entire simulation stays within the realm of a normal gait.

V. Basin of Attraction

To ensure a steady gait, it is crucial to refine our walking model, which involves determining the initial equation value, or in this case, the value of y0. According to the equation (3.1) and (3.2) we understand the relationships $\phi = 2\theta$ and $\dot{\phi} = \dot{\theta}(1 - \cos(2\dot{\theta}))$, but the values of θ and $\dot{\theta}$ are not clear. Therefore, we resort to the 'Basin of Attraction'[1] methodology to ascertain these initial values of our Ordinary Differential Equations (ODEs). Essentially, the basin of attraction delineates the spectrum of states that yield a viable and stable walking pattern.

A. Establishing the Basin of Attraction Function

The procedure for setting up the Basin of Attraction function is outlined as follows:

1) Define the initial value range for θ and $\dot{\theta}$.
2) Construct a matrix to store the stable walking results pertaining to θ and $\dot{\theta}$.
3) Establish two loops to compute all possible combinations of θ and $\dot{\theta}$ within the specified range.
4) Utilize the primary loop function from the previous section to solve the ODEs.
5) Extract the results and generate an illustrative image of the Basin of Attraction.

For the initial value range for θ and $\dot{\theta}$, after several times of simulation, firstly the set range are as follows[1]:

$$\theta \in [0, 1]$$
$$\dot{\theta} \in [-1.2, 0]$$

B. The Application of the Basin of Attraction function

After establishing the Basin of Attraction function, we can glean information regarding the stability of the model and the number of steps it can take.

Initially, we feed the parameters of θ and $\dot{\theta}$ into the main loop to perform the simulation. The output gleaned from this loop provides us with crucial details; when the model walks 'normally' or 'almost normally', it conveys information that includes the number of steps taken.

The 'imagesc' function in Matlab serves as an excellent tool to display the Basin of Attraction. It indicates the points at which the model can walk and uses colors to denote the number of steps. In our scenario, 'yellow' signifies that the model can walk 5 steps, while 'orange' indicates that it can walk 4 steps.

As depicted in the figure 4, the horizontal axis represents θ, the vertical axis on the left signifies $\dot{\theta}$, and the color indicates the number of steps taken by the model.

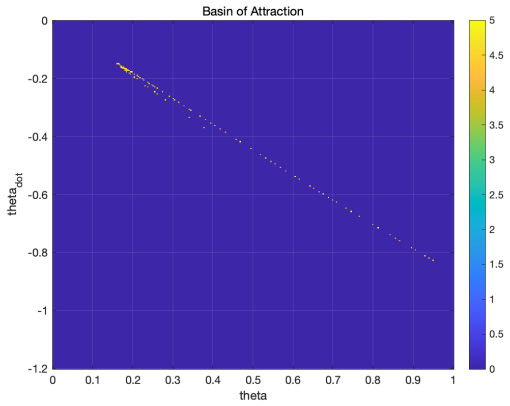

Fig. 4. Basin of Attraction

VI. Conclusion

As per the requirements, we sequentially simulate the model when γ is within the range of [0.002,0.008], with a step size of 0.002, as depicted in the figure 5:

From this, we can observe that the walking model can achieve stability when the values of θ and $\dot{\theta}$ correspond to the

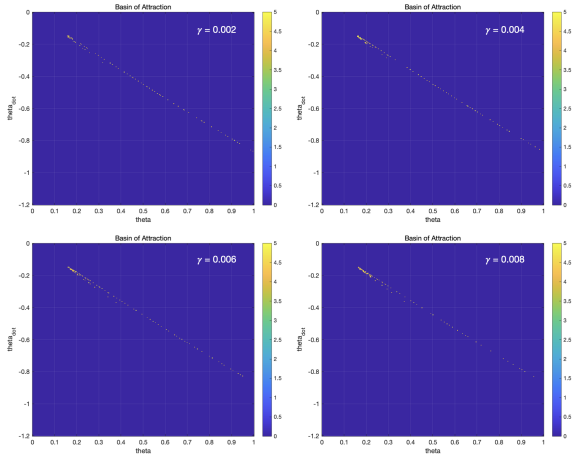

Fig. 5. Contrast of Basin of Attraction

yellow region in the image. In all the blue areas, the model cannot maintain stable walking.

As the angle γ increases, we can see that the initial Basin of Attraction concentrates around the point (0.2, -0.2). Subsequently, we narrow the interval to the following range:

$$\theta \in [0.15, 0.25]$$
$$\dot{\theta} \in [-0.26, -0.14]$$

Continuing the simulation and analysis on this area for comparison, we obtained the figure 6:

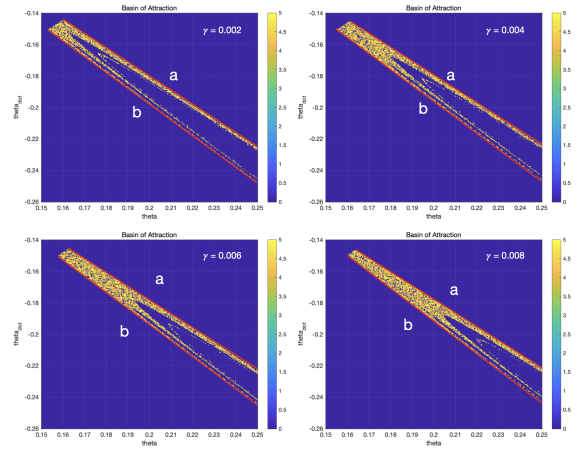

Fig. 6. Contrast of Basin of Attraction

This indicates that as the slope increases, the distance between line a and line b becomes smaller. This implies a higher correlation between θ and $\dot{\theta}$. Furthermore, when θ is less than 0.21, the distribution of the model's stable initial value solutions becomes denser. This also suggests that when the angle γ is larger, the distribution of the model's stable initial value solutions becomes more concentrated. In other words, when within the range

$$\theta \in [0.15, 0.25]$$
$$\dot{\theta} \in [-0.26, -0.14]$$

the model's stability improves as γ increases.

Currently, we have analyzed the influence of different inclinations of slopes on stability. In the future, further insights can be gained by exploring variations in the basin of attraction under different parameters. This exploration could help determine which parameters affect the stability of the walking model, such as factors like body weight and leg length. Uncovering the underlying patterns among these parameters can be achieved through the establishment of the basin of attraction. Additionally, an investigation into the tipping tendencies during instability could reveal factors that lead the model to lean forward or backward.

Moreover, a more in-depth inquiry into the impact of structure on the stability of the walking model is possible. For instance, recognizing the similarity in properties between our tendons and springs allows for the integration and optimized analysis of our walking model with the Spring-Loaded Inverted Pendulum (SLIP) model. This fusion can potentially enhance the model's capacity to emulate human motion in certain performance aspects. The parameters and patterns elucidated through these investigations could serve as rules for optimizing robotic motion in the future.

REFERENCES

[1] Martijn Wisse, Arend L Schwab, Richard Quint van der Linde, and Frans CT van der Helm. How to keep from falling forward: Elementary swing leg action for passive dynamic walkers. *IEEE Transactions on robotics*, 21(3):393–401, 2005.

[2] Mario W Gomes. A derivation of the transisition rule at heelstrike which appears in the paper "the simplest walking model: Stability, complexity, and scaling" by garcia et al. *Mariano, et al*, 1999.

[3] Mariano Garcia, Anindya Chatterjee, Andy Ruina, and Michael Coleman. The simplest walking model: stability, complexity, and scaling. 1998.