DP 2: Differential-Drive Robot in Artificial Gravity

Tamer Saatci[1] and Conan Zhang[2]
University of Illinois Urbana-Champaign, Champaign, Illinois, 61820, United States

This report describes the design of a control system behind a differential-drive robot that travels inside a circular space station. A state-space model is used to linearize a closed feedback loop between the system and robot. The theory behind the model is described, and the system requirements, with their verification methods, are specified. Results of the experimentation are provided along with a relevant discussion.

I. Nomenclature

A	=	state matrix
B	=	input matrix
e_{lat}	=	lateral position error, m
K	=	gain matrix
Q	=	state cost matrix
R	=	cost control matrix
t	=	time, s
τ_l	=	left wheel torque, Nm
τ_r	=	right wheel torque, Nm
θ	=	pitch angle, rad
u	=	input
v	=	segbot velocity, m/s
$v_{station}$	=	space station rotational speed, rad/s
W	=	controllability matrix
x	=	state

II. Introduction

Robots are a promising source of innovation for the aerospace industry due to their unmanned and robust capabilities. Although design elements are key to a successful robot, its usability plays a critical role in many real-world applications. This report details the development of a segbot, a differential-drive robot intended to travel within a rotating circular space station under artificial gravity. As the segbot navigates using its two independent wheels, the system applies a torque to stabilize its pitch and velocity. This report's objective is to thus explain the steps towards developing an optimal control system that allows a user who is not an engineer to control the segbot's lateral location while keeping it upright. For this design, a reliable controller should be developed, and the system should meticulously be tested to verify its safety requirements.

III. Theory

A. State Space Model

To build an optimal controller, the dynamical system of equations for the segbot are placed into a linear state-space model consisting of ordinary differential equations (ODEs). Through linearization, the complex equations of

[1] Tamer Saatci, Aerospace Engineering
[2] Conan Zhang, Aerospace Engineering

motion of the system can be simplified, enabling more thorough analysis on the system's behavior under varying conditions.

$$\dot{x} = Ax + Bu \tag{1}$$

The first step involves deriving the equations of motions for the segbot. Using the approach in classical mechanics, the resulting system model can be depicted in Eq 2.

$$\begin{bmatrix} v\sin(\phi) \\ \frac{-18.4\tau_l - 18.4\tau_r - 14.4\left(\dot{\phi}^2+\dot{\theta}^2\right)\sin(\theta) + 37.4\cdot\left(0.48\dot{\phi}^2\sin(2\theta) - \tau_l - \tau_r + 23.5\sin(\theta)\right)\cos(\theta)}{89.9\cos^2(\theta) - 93.5} \\ \frac{-1.06\dot{\phi}\dot{\theta}\sin(2\theta) - 2.4\dot{\phi}v\sin(\theta) - 1.08\tau_l + 1.08\tau_r}{0.96\sin^2(\theta) + 1.03} \\ \frac{-7.49\dot{\phi}^2\sin(2\theta) + 15.6\tau_l + 15.6\tau_r + 2.4\cdot\left(3.08\tau_l + 3.08\tau_r + 2.4\left(\dot{\phi}^2+\dot{\theta}^2\right)\sin(\theta)\right)\cos(\theta) - 367.0\sin(\theta)}{5.76\cos^2(\theta) - 5.99} \end{bmatrix} \tag{2}$$

Replacing $\dot{\theta}$ with ω_θ and $\dot{\phi}$ with ω_ϕ, the initial 2nd order system of equations of motion for the segbot can be broken down into a set of first-order ODEs. Two ODEs are then added to the function to account for the two new replacement variables. The system in Eq. 3 can now begin the linearization process.

$$\begin{bmatrix} \dot{e_{lat}} \\ \dot{v} \\ \dot{\omega_\phi} \\ \dot{\omega_\theta} \\ \dot{\theta} \\ \dot{\phi} \end{bmatrix} = f(e_{lat}, v, \phi, \omega_\phi, \theta, \omega_\theta, \tau_l, \tau_r) = \begin{bmatrix} v\sin(\phi) \\ \frac{-18.4\tau_l - 18.4\tau_r - 14.4\left(\omega_\phi^2+\omega_\theta^2\right)\sin(\theta) + 37.4\cdot\left(0.48\omega_\phi^2\sin(2\theta) - \tau_l - \tau_r + 23.5\sin(\theta)\right)\cos(\theta)}{89.9\cos^2(\theta) - 93.5} \\ \frac{-1.06\omega_\phi\omega_\theta\sin(2\theta) - 2.4\omega_\phi v\sin(\theta) - 1.08\tau_l + 1.08\tau_r}{0.96\sin^2(\theta) + 1.03} \\ \frac{-7.49\omega_\phi^2\sin(2\theta) + 15.6\tau_l + 15.6\tau_r + 2.4\cdot\left(3.08\tau_l + 3.08\tau_r + 2.4\left(\omega_\phi^2+\omega_\theta^2\right)\sin(\theta)\right)\cos(\theta) - 367.0\sin(\theta)}{5.76\cos^2(\theta) - 5.99} \\ \omega_\theta \\ \omega_\phi \end{bmatrix} \tag{3}$$

B. Equilibrium Points

Setting the left side of Eq. 3 to 0, each of the variables can now hold a value to make the system reach equilibrium. Values of 0 for all variables except for the segbot velocity were chosen to stabilize the state-space system as shown in Eq. 4, improving the system's response around a moving segbot.

$$\begin{gathered} e_{lat}, \phi, \omega_\phi, \theta, \omega_\theta, \tau_l, \tau_r = 0 \\ v = 0.4 \end{gathered} \tag{4}$$

The non-zero velocity component was also chosen as an equilibrium point due to its necessity in keeping the controller controllable. When using zeroed out values, the controller would inherently "break," not functioning properly during certain simulations. While any value of segbot velocity (if all other variables are 0) could be used as an equilibrium point, a velocity of 0.4 m/s was chosen for the simulations to keep the robot at a safe speed and within the realm of controllable operation.

C. Derivation of A & B Matrices

The next step is to place the space-space model into state feedback form. By taking the Jacobian of $\dot{x}$ with respect to the state variables m (e_{lat},v ,ϕ ,ω_ϕ,θ ,ω_θ) and the control inputs n (τ_l,τ_r), coefficient matrices A and B can be

found. The result is a linear approximation of the right-hand side of the first ODEs. Plugging in the equilibrium points found earlier, a linearized state-space model is derived.

$$A = \frac{\partial f}{\partial m} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0.4 \\ 0 & 0 & 0 & 0 & -245.06 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1592.92 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \quad B = \frac{\partial f}{\partial n} = \begin{bmatrix} 0 & 0 \\ 15.53 & 15.53 \\ -1.05 & 1.05 \\ -99.68 & -99.68 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \tag{5}$$

D. Controllability

To test controllability, *W* can be derived from *A* and *B*. *W* is a *n* x (*n***m*) matrix where *n* is the number of columns of *A* and *m* is the number of columns of *B*. Each column of W is obtained through matrix multiplication A^iB where *i* is the order of column between 0 and *n*-1.

$$W = [B \; AB \; \ldots \; A^{n-1}B] \tag{6}$$

If the rank of *W* is equal to *n*, then every state of the system can be influenced by the control inputs, meaning the system is fully controllable. Computing *W* with the coefficient matrices *A* and *B* in Eq. 5, *rank*(*W*) = 6 = *n*, the exact amount states. The initial conditions were chosen so that the system would satisfy controllability.

E. Linear Quadratic Regulator (LQR) Method

Using the LQR Method, gain matrix *K* can be found to simplify the state-space model into a solvable linear system for a specific choice of inputs known as state feedback *u*. Choosing *u* = -*Kx*, leads to a system depicted in Eq. 7. For the controller to be asymptotically stable, all eigenvalues of (*A* – *BK*) must have a negative real part.

$$\begin{aligned} \dot{x} &= Ax + Bu \\ &= Ax + B(-Kx) \\ &= (A - BK)x \end{aligned} \tag{7}$$

The LQR method allows for prioritizing certain weights of specific variables in the state-space model, while also holding the system to negative eigenvalues. *Q* is a symmetric matrix where the diagonal elements correspond to the system's state variables *m*. *R* is also a symmetric matrix where the diagonal elements correspond to each control input *n* in the system. Every diagonal element corresponds to the respective state or input variable, indicating which variable yields greater importance in converging towards equilibrium state.

$$Q = \begin{bmatrix} 0.1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 100 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.1 \end{bmatrix} \tag{8}$$

$$R = \begin{bmatrix} 0.1 & 0 \\ 0 & 0.1 \end{bmatrix}$$

After multiple trials of altering the weight of one variable at a time in *Q* and *R* from and at a time while holding all other variables constant, pitch angle θ was deemed to have the greatest weight in keeping the segbot stable. The experimentation and results stay consistent with the notion that pitch angle is what keeps the segbot upright when

translating in motion. This preliminary observation was crucial in determining the optimal gain matrix towards computing the control inputs. The resulting *Q* and *R* matrices can be seen in Eq. 8.

$$A^T P + PA - PBR^{-1}B^T P + Q = 0 \tag{9}$$

Solving the continuous-time algebraic Riccati equation in Eq. 9 and the gain matrix equation in Eq. 10, yields a *K* that minimizes the cost function and achieves desirable controller conditions.

$$K = R^{-1}B^T P =$$

$$K = \begin{bmatrix} -0.707 & -0.707 & -2.514 & -2.419 & -31.961 & -1.386 \\ 0.707 & -0.707 & 2.514 & -2.419 & -31.961 & 1.386 \end{bmatrix} \tag{10}$$

IV. Experimental Methods

To establish performance limits for the controller, two requirements must be met by two verifications. The following section details the parameters, requirements, verifications, and trial procedures for the experiment.

I. First Requirement

The segbot shall reach a lateral error between -0.3 meters and 0.3 meters in under 10 seconds from any initial lateral error of -1.5 meters to 1.5 meters. The segbot should satisfy this requirement if it is expected to be used practically, since there exist many scenarios where a robot needs to reach a target location while initially being at a non-centralized location. Thus, it is important to ensure that the segbot can operate under a wide range of initial lateral positions correcting itself to a more stable range. Here, stability is defined as achieving the lateral error range of (-0.3,0.3). Additionally, an optimal controller should bring the system quickly to equilibrium. Ensuring the segbot can correct its lateral error to a stable equilibrium level in less than 10 seconds adds another level of reliability to the controller.

II. Second Requirement

The segbot shall remain vertically upright while manually commanded to any lateral position between -1.5 meters and 1.5 meters in a 15 second interval. The segbot should satisfy this requirement if it is expected to be controlled by an external user who is not an engineer. The segbot must be able to deviate from the center of the ring if needed without losing its stability and should reach the desired destination in a controlled amount of time. Here, being vertically upright is defined as a non-zero pitch angle range of (0.0, 2.0) degrees, indicative of the segbot not falling over and continuing in motion. Ensuring that the segbot can reach any location in a fifteen second interval suggests a robust and well-built controller.

III. Verification

To verify the first requirement, the segbot underwent multiple simulation instances at varying initial lateral error values up to ±1.5 meters with its behavior being observed through the lateral error vs. time graph. Whether the lateral error could return to between 0.3 meters and -0.3 meters in under ten seconds was verified through testing. The initial lateral error would first be chosen as equilibrium (0 meters), increasing/decreasing by 0.5 meters to the max range after each successful trial.

To verify the second requirement, the simulation environment started with a zero-target lateral position. Next, the segbot was commanded to move in any direction of any lateral position between –1.5 meters and 1.5 meters using manual inputs provided by the simulator. At fifteen seconds, if the segbot pitch angle is between 0° and 2°, then the requirement is met. During all trials, the initial conditions outlined in Eq. 11 were kept the same to constrain the experiment to the same simulation parameters.

$$\begin{bmatrix} v_{station} \\ t_{max} \end{bmatrix} = \begin{bmatrix} -0.1 \\ 15 \end{bmatrix} \tag{11}$$

IV. Trials

The segbot's behavior was tested in the PyBullet simulation through the Jupyter Notebook environment. The simulation generated the segbot moving on the inner surface of a ring. Keyboard commands could be used to direct the segbot's lateral location. The simulation recorded the segbot's lateral position, target lateral position, lateral error, velocity, heading angle, heading rate, pitch angle, pitch rate, torque applied on the left wheel, & torque applied on the right wheel all with respect to time. These plots were displayed in graphs through Python package *matplotlib.pyplot.*

V. Results & Discussion

To verify the first requirement, the segbot was tested with 9 different initial lateral error values ranging from -1.72 meters to 1.72 meters, where ±1.72 meters was determined to be the controller's maximum initial lateral error before the segbot fell off the inner ring while trying to re-align itself. This range is greater than the range stated in the requirement. The data for the tests were recorded for 15 seconds and are shown in Fig. 1. The segbot's lateral error was observed to have reached the desired threshold between –0.3 meters and 0.3 meters in less than 10 seconds.

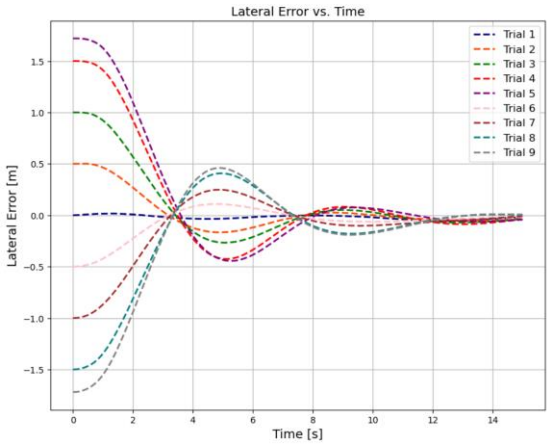

Fig 1. Lateral Error vs. Time Graph

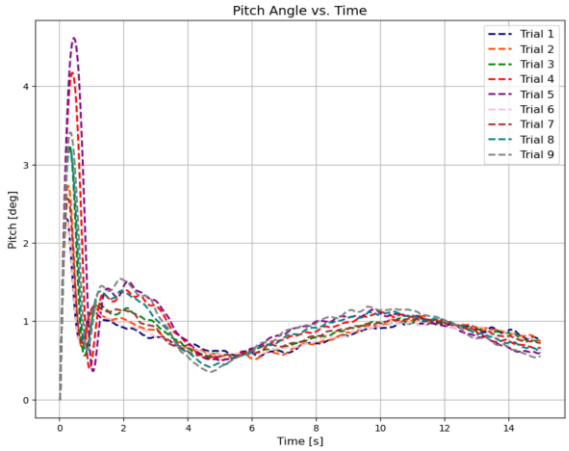

Fig 2. Pitch Angle vs Time Graph

Additionally, in Fig. 2, the segbot is observed to have maintained an upright vertical state throughout each trial, suggesting the segbot was moving in a stable manner. Therefore, the first requirement is verified. Initial and final lateral error data for the positive lateral errors are also listed in Table 1 with a 0.01-meter margin of error for the curious eye.

Trial	Initial Lateral Error [m]	Final Lateral Error [m]
1	0.00	-0.01
2	0.50	-0.02
3	1.00	-0.02
4	1.50	-0.02
5	1.72	-0.02

Table 1 First Requirement Verification Data

To verify the second requirement, the segbot was tested in 5 new trials with target lateral position values ranging from -1.5 meters to 1.5 meters. The segbot was commanded to reach an arbitrary target lateral position using keyboard inputs [(*j*) - move the target to the left] or [(*l*) – move the target to the right]. In trials 12 and 14, the segbot was additionally moved for a second time to another target location, adding another level of difficulty towards verifying the second requirement. The data for the tests were recorded for 15 seconds and are shown in Fig. 3.

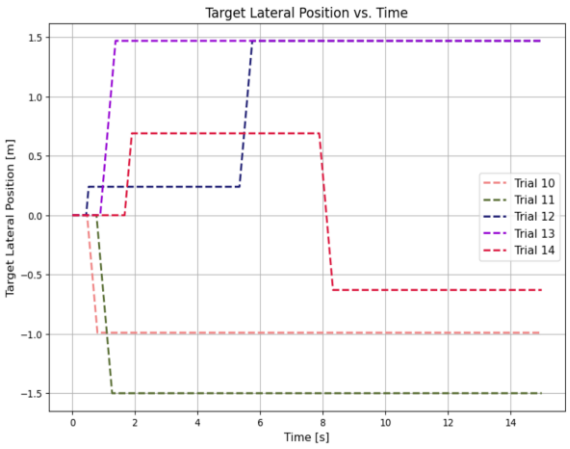

Fig 3. Target Lateral Position vs. Time Graph

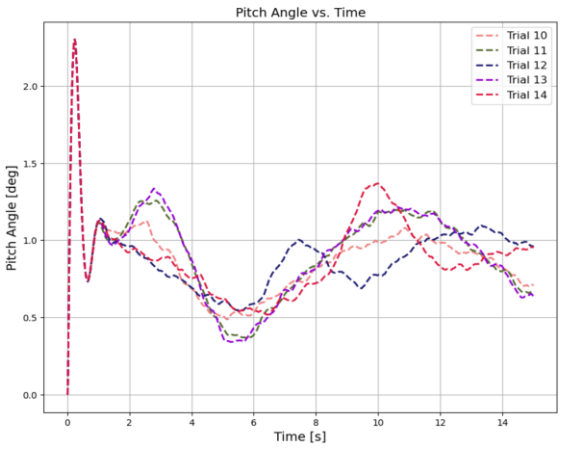

Fig 4. Pitch Angle vs Time Graph

In Fig. 4, the segbot's pitch angle was observed to have a pitch angle range between 0° and 2° at the 15 second mark in all 5 trials. In trials 11, 12, and 13, the max ranges of –1.5 meters and 1.5 meters were reached. Thus, the segbot maintained an upright vertical state within ±1.5 meters. Hence, the second requirement is verified. Data with a 0.02-unit margin of error regarding the target lateral position for the segbot's movement and the final pitch angle are listed in Table 2 for the curious eye.

Trial	Target Lateral Position [m]	Final Pitch Angle [deg]	Time Vertically Upright [s]
10	0.00 → -1.00	0.70	15
11	0.00 → -1.50	0.60	15
12	0.00 → 0.25 → 1.50	0.95	15
13	0.00 → 1.50	0.60	15
14	0.00 → 0.70 → -0.65	0.95	15

Table 2 Second Requirement Verification Data

VI. Conclusion

This paper discusses the implementation of a control system for a segbot which can either move autonomously or by human control on a circular spaceship generating artificial gravity. The controller's success can be attributed to the Linear Quadratic Regulator method (LQR). Through LQR, the segbot prioritized its stability with a heavy emphasis on pitch angle over all other state and input variables. To ensure that the segbot could properly function at a high level of performance and with human intervention, several trials were conducted under varying conditions within a simulation environment. For requirement one, the trials verified that the robot could centralize around a wide range of lateral positions on the spaceship with its final lateral error between (-0.3, 0.3) meters. For requirement two, the trials verified that the robot could remain vertically stabilized while moving to a wide range of positions on the spaceship through human command with its final pitch angle range between (0.0, 2.0) degrees. To expand on this experiment, future studies could be tested on the same trials, but with a more optimized Q and R matrix-weight choices to push the boundary limits of lateral error and controller speed, improving the overall efficiency of the control system.

Acknowledgments

A special thanks to Graduate Assistant Pedro Leite for his comments and suggestions on the initial draft. His insight provided much help towards improving the Theory and Experimental Methods section.

References

"Chang, W. (2024), "Example Reports-Students," AE 353: Aerospace Control Systems Available: https://uofi.app.box.com/s/a58fdbijnzcxs43pvchmc1cy8tfcot52

Appendix

A. Reflection

We were able to communicate effectively over SMS while file sharing and relaying information over Microsoft Teams. We kept in touch regularly to check up on each other, provide feedback, and offer ideas that could be implemented into our controller design. We communicated thoroughly over splitting and delegating tasks, as the entire report was completed effectively and timely.

B. Work Log

Day	Task	Person
2/25	Microsoft Teams, Word, and Draft Setup	Tamer & Conan
2/25	Linearized system to find A and B Matrices	Tamer & Conan
2/25	Wrote first draft of Theory	Conan
2/25	Wrote first draft of Experimental Methods	Tamer
2/26	Wrote about the math behind Theory	Conan
2/26	Wrote about Verification methods for requirements	Tamer
2/27	Tested Q and R matrices for ideal K matrix	Conan
2/28	Made final changes on the document for the draft submission	Tamer & Conan
3/4	Updated Theory formatting	Conan
3/5	Wrote Introduction and Abstract	Tamer
3/6	Updated LQR and State-Space Model sections	Conan
3/6	Updated Theory and Experimental Methods	Tamer
3/7	Developed code for Results & Discussion	Conan
3/7	Wrote analysis for Results & Discussion	Tamer
3/8	Marked up Code	Conan
3/8	Added Conclusion	Tamer
3/8	Final Formatting	Tamer & Conan