Spacecraft Optimal Docking Procedure Using LQR and Moving Targets

Derald Madson III
Dept. of Aerospace Engineering
Univerisity of Illinois
Urbana Champaign, Illinois

Andrew Pelster
Dept. of Aerospace Engineering
Univerisity of Illinois
Urbana Champaign, Illinois

Sanjana Srivastava
Dept. of Aerospace Engineering
Univerisity of Illinois
Urbana Champaign, Illinois

***Abstract*— This report describes the design and implementation of a controller for spacecraft docking. The controller utilized in this problem is a linear-quadratic regulator (LQR) and controls the position and angle coordinates of the spacecraft in the system. Several situations are described for the spacecraft and docking port position and motion, and it was found that the spacecraft was able to reach the desired distance tolerance between itself and the docking port.**

I. Problem Motivation

Spacecraft docking is a specific rendezvous maneuver which involves the temporary or permanent joining of two individual free-flying vehicles in outer space. The first development of a spacecraft docking maneuver was with NASA's Project Gemini and was first successfully performed manually under command of Neil Armstrong for Gemini 8 in 1966. Since then, spacecraft docking has been a necessary maneuver for many further spaceflight missions like the Apollo program for moon landings, using piloted manual docking. However, the Soviet Union employed autonomous docking from the beginning of its docking endeavors, with two uncrewed Soyuz vehicles successfully docking in 1967. For two spacecraft to be able to rendezvous autonomously in space, the applications into advanced spaceflight operations would span a wide range of applications. Autonomous docking would aid in servicing missions, emergency repairs, and even to prepare human spaceships before crew arrival, building an efficient and faster transit to and from Mars. It also has vital applications in in-orbit assembly of large units, crew exchange, and refueling and retrieval operations.

Docking systems are classified as 'hard' or 'soft' dockings based on the relative speed of the vehicles. A docking maneuver consists of a chaser vehicle to contact a stationary target vehicle within the boundaries of velocity, position, attitude, and angular rate. The state parameters of the vehicle will be controlled by the GNC system of the chaser to enter the docking interface. For a linearized spacecraft system, the use of a Linear Quadratic Regulator (LQR) control has proven to be advantageous to minimize the control energy required and optimize response performance. LQR control law also aids in improved overall control with lower errors as compared to other control designs like the proportional-derivate (PD) method.

Spacecraft docking using an integral LQR has been previously investigated by Nandagopal et al. [1], where translational motion was defined for spacecraft rendezvous and docking using the Clohessy-Wintshire (CW) equations; and feedback control loops implemented through gas jet thrusters. Guarnaccia et al. [2] extended the use of LQR for full 6-DOF motion control of a spacecraft based on a state dependent Riccati equation. The paper accounts for both position and attitude and quantifies the trade-off between feasibility and optimality in real-time. Yang et al. [3] proved the efficacy of using LQR with a quaternion-based method to design nonlinear spacecraft control. The paper shows the ease of formulating an analytical LQR controller and establishes design parameters and closed loop eigenvalues interactions. The results show that the LQR controller improves global stability of the spacecraft system. A comparison between LQR and PD methods was carried out by Beatty [4], comparing optimal gains of LQR with non-optimal gains of PD to improve performance for a spacecraft orbiting Earth in a Molniya orbit. Comparing step responses showed reduced angular momentum and subsequently smaller wheel size for LQR control. Starin et al. [5] further show that use of LQR control reduces weight of the propulsion system for spacecraft formation flying. Closed loop simulations demonstrated effective control and maneuverability within mission boundaries.

In this paper we test an LQR docking system against increasingly complex scenarios and compare it to more modern methods. By using LQR we seek to design a computationally cheap controller that operates stably, safely, and optimally. Based on our experimental scenarios built in MATLAB we demonstrate the usefulness of LQR in rendezvous and docking maneuvers.

II. PROBLEM FORMULATION

To start solving the problem, we developed a state space model that represents the dynamics of a system.

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

$$\begin{bmatrix}\dot{x}\\ \dot{y}\\ \dot{z}\\ \dot{\alpha}\\ \dot{\beta}\\ \dot{\gamma}\end{bmatrix} = \begin{bmatrix}1 & 0 & 0 & -v_{i-1}sin(\alpha)dt & 0 & 0\\ 0 & 1 & 0 & 0 & -v_{i-1}sin(\beta)dt & 0\\ 0 & 0 & 1 & 0 & 0 & -v_{i-1}sin(\gamma)dt\\ 0 & 0 & 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 0 & 0 & 1\end{bmatrix}\begin{bmatrix}x\\ y\\ z\\ \alpha\\ \beta\\ \gamma\end{bmatrix} +$$

$$\begin{bmatrix}\dot{x}\\ \dot{y}\\ \dot{z}\\ \dot{\alpha}\\ \dot{\beta}\\ \dot{\gamma}\end{bmatrix}\begin{bmatrix}0 & 0 & 0 & 0 & 0 & \dot{x}\\ 0 & 0 & 0 & 0 & \dot{y} & 0\\ 0 & 0 & 0 & \dot{z} & 0 & 0\\ 0 & 0 & dt & 0 & 0 & 0\\ 0 & dt & 0 & 0 & 0 & 0\\ dt & 0 & 0 & 0 & 0 & 0\end{bmatrix} \quad (2)$$

In these equations, equation (1) is the variable states and $\dot{x}$ is the rate of change of the states, while u is the control input. The states to control are the three position coordinates, x, y, z and the three angular coordinates, α, β, γ. To implement control, a linear quadratic regulator (LQR) controller was used. The equations that describe it are seen below.

$$Q = \begin{bmatrix}1 & 0 & 0 & 0 & 0 & 0\\ 0 & 1 & 0 & 0 & 0 & 0\\ 0 & 0 & 1 & 0 & 0 & 0\\ 0 & 0 & 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 0 & 0 & 1\end{bmatrix} R = \begin{bmatrix}1 & 0 & 0 & 0 & 0 & 0\\ 0 & 1 & 0 & 0 & 0 & 0\\ 0 & 0 & 1 & 0 & 0 & 0\\ 0 & 0 & 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 0 & 0 & 1\end{bmatrix} \quad (3)$$

$$x_N^T Q x_N + \sum_{k=0}^{N-1} (x_k^T Q x_k + u_k^T R u_k) \quad (4)$$

$$P_{k-1} = A^T P_k A - (A^T P_k B)(R + B^T P_k B)^{-1}(B^T P_k A) + Q$$
$$K_n = (R + B^T P_{n+1} B)^{-1}(B^T P_{n+1} A) \quad (5)$$

Equation (4) describes the cost function associated with this controller where Q and R are weighted matrices and depend quadratically on the state and control variables.

For this application, the control was split up into three different sections. First, the spacecraft was aligned with the docking port and started the approach. Then the angles were adjusted so each of the angles are correct to line up with the docking port angles. Finally, the spacecraft made the final approach to arrive at the docking port.

To simulate this problem, the computer program MATLAB was used. The equations seen above were implemented in MATLAB and iterated until all the position and angle states approached a value of 0 relative to the docking port.

For the initial conditions of the spacecraft, they were randomized to test the controller for a variety of situations. Additional components that made the problem more interesting were having the docking port moving and ensuring the spacecraft can still approach the docking port.

With this problem, it is important to constrain the spacecraft to not go beyond the position of the docking port, otherwise this would cause a crash in real life, and to approach a velocity of 0 when each of the states approach the value of the docking port's states. Also, a position tolerance was added, because in a real-life situation, the operators might want a spacecraft to stop just before the docking port to confirm everything is lined up correctly. Another potential reason to leave space in one of the axes is if the mechanism for docking is magnetic, so the very last step can be done without the controller. This constraint can easily be removed or adjusted based on the docking port scenario by setting the labeled MATLAB variable in our simulation code.

Three different combinations of the LQR controller were created in this problem to iterate the problem. The first LQR controller, designed bring the chaser to close approach with the target, used 3 DOF to control the position states and had the following Q matrix, while the R was a 3 x 3 identity matrix.

$$Q_1 = \begin{bmatrix}1 & 0 & 0\\ 0 & 10 & 0\\ 0 & 0 & 10\end{bmatrix} \quad (3)$$

For the second controller, which controls angular alignment with 3 DOF, the Q and R matrices were both 3 x 3 identity matrices. The LQR function does not proceed to the next stage until the chaser docking port is aligned angularly with the target docking port.

For the final controller, designed to move the chaser vehicle slowly and safely into the acceptable docking position, both positions and angles using 6 DOF were adjusted so the Q and R matrices took the form shown in equation (3).

Another aspect of this problem was the position of the docking port. We created three individual scenarios and within each tested the LQR docking algorithm multiple times. In the Conclusion section we provide a demo of each scenario. For *Scenario 1*, the controller was tested with a stationary non-rotating docking port at the origin. This is a valid orbital docking scenario when both the chaser and the target are on a similar orbit and standard maneuvering techniques are utilized to get the chaser within docking range. In *Scenario 1* we made the preceding assumption. In *Scenario 2* the docking port was given a complex known periodical trajectory, this simulates a random perturbation on target object which could be caused by multiple factors. We then calculate the optimal trajectory using multiple LQR scenarios and utilize the computed trajectory that creates the least time cost. *Scenario 3* is similar to *Scenario 2*, as the docking port still moves on a complex trajectory but is also rotating randomly. This simulates a tumbling target.

Furthermore, we design two different LQR methods for the initial stage. The two different cases were evaluated and shown in the Conclusions section. By manipulating the LQR formula we create significantly different approaches for safety reasons as described in the Challenges section.

Several constraints to ensure both realism and safety were added. The previously mentioned staging and final docking distance constraint, a limit of velocity in the initial and final LQR sections, a rotational limit on the chaser. Adding these constraints increases complexity and realism of the LQR rendezvous algorithm.

III. Challenges

There were several challenges encountered by the research team when designing an optimal LQR based docking controller. The most obvious challenge was determining how to find the most optimal course without colliding with the docking port or target. In our research and computational experimentation, we implemented real world limits and safety measures. Therefore, the primary challenge to be resolved was balancing the constraints implemented for realism and safety reasons and the optimality of the LQR controller. Also, because we wanted to ensure the controller, while still performing optimally, could be generalized for multiple purposes such as near rendezvous and docking we had to design a controller that was adaptable and would not fail in various scenarios. Finally, since we desired to create a fast, yet still accurate controller, we need to reduce computation time without sacrificing optimality.

A. *Optimal Trajectory vs. Safe Approach*

Trivially, one can predict the most optimal course for docking with a non-moving target is to fly straight towards the target, whilst at the same time achieving the proper alignment, and stopping just moments before the objective. This method, in addition to being unsafe, is also only true when the spacecraft can see the target docking port. This is made clear by Fig. 1. In the first image of Fig. 1 the blue chaser spacecraft can see the target red docking port and theoretically could fly directly to the port in the trivial manner described above. In the second image the spacecraft cannot fly straight to the docking port and either crashes into it or must maneuver around it. The simplest resolution to this challenge is for the spacecraft, instead of maneuvering directly to the target object, maneuvers to a point along the targets docking port axis. This is precisely why we implemented our three-stage docking method discussed in Problem Formulation.

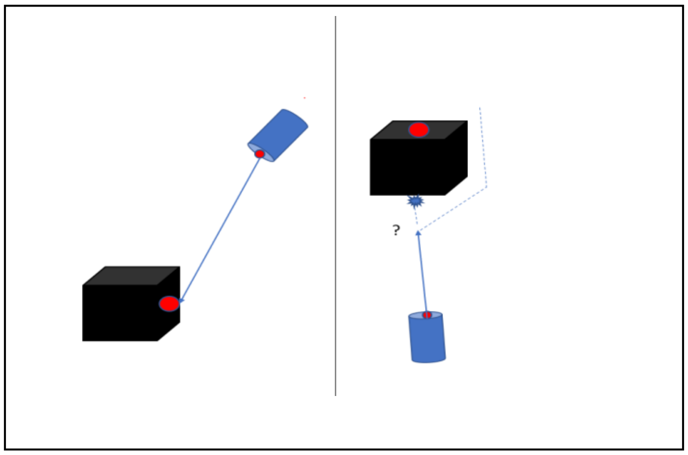

Fig. 1. Trivial ideal (left) vs reality (right)

To meet the challenge of simulating real world conditions, we designed all three phases to limit risk and stress on the spacecraft by limiting its velocity. In particular, we limit the spacecrafts total velocity relative to the target in the final docking stage to reduce the risk of catastrophic collision.

Additionally, our controller was also programmed to be easily adaptable so that the initial phase target can be easily changed. For example, a manned mission may be more unwilling to pass closely to the target than a drone cleaning space debris. The overall design behind the LQR controller sacrifices small amounts of optimality for safety.

Note that while the LQR controller does avoid collisions with the target object in its initial phase, and velocity constraints are implemented in all phases we have not implemented rigorous safety collision-avoidance control algorithm on an advanced level similar those proposed by Dong or Xu [6] [7].

B. *Adaptability*

To meet the intent of creating an adaptable and optimized docking system with LQR, we ensured the chaser could execute multiple docking procedures based on target conditions. For example, the spacecraft may desire to approach the target along its docking axis for an extended period, whereas other docking purposes may desire the shortest path to the target without colliding with it. We met this challenge by adaptively modifying our LQR algorithm based on user selected variables and dynamically monitoring the spacecrafts position relative to the target. When the position of the spacecraft and docking port are within certain parameters, we change our LQR problem set to create the desired approach path. Both Fig. 3 and Fig. 4 in the result section illustrate how we implemented the dynamic LQR adaptations. Fig. 2 also shows an example of a dynamic change in the LQR calculations as the spacecraft approaches the target.

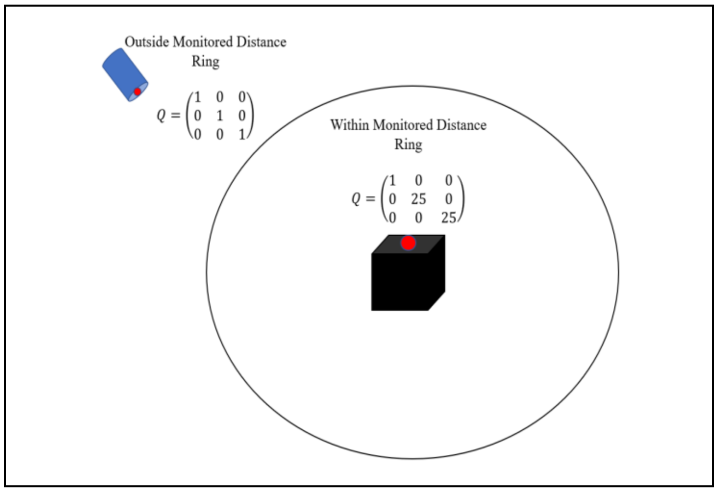
Outside Monitored Distance Ring
$Q = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$
Within Monitored Distance Ring
$Q = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 25 & 0 \\ 0 & 0 & 25 \end{pmatrix}$

Fig. 2. Adaptive LQR, (As the spacecraft nears the target the LQR iterations change the path of the spacecraft to establish the desired trajectory)

The LQR function can also adaptively modify its state space model, specifically, modifying the B matrix to rapidly change how the controller implements control commands. This functionality is specifically used to ‘shut off’ motion along the docking axis of the target when the spacecraft is close enough.

Further implementation can also use this method to dynamically avoid debris.

C. Computation Time

a) Non-Moving Target For a non-moving target the computational expense is very small since only one LQR optimal control path must be calculated. This is discussed further in the findings section.

b) Moving Target We found that computing the optimal trajectory for a moving target was significantly more computationally expensive. Although the LQR controller is fast at computing each individual trajectory; computing mutliple trajectories for a moving target and then deciding which one is optimal takes a significant amount of time. Once the spacecraft decides the path to take computations for each control step would proceed quickly. However, the initial calculation takes an extremly long amount of time. The amount of time required can be reduced by simplifying the problem. Additionally, we could have reduced computation time by analyzing less possible routes to the target, but this jeopordized the optimtimization. We ultimately did not discover a sufficient way to reduce the computational time for optimal LQR docking on a moving target and discuss further in the findings section.

IV. Findings

During multiple simulations we found that for docking and rendezvous, LQR is an effective controller, rapidly converges to the optimal u^*, and can safely control the docking procedure. When the randomness or absolute velocity of the motion of the target is increased LQR generally becomes less effective and reasonable timing is more difficult to achieve.

A. Results of three distinct simulation scenarios:

1) We use an LQR controller to simulate rendevous and docking with a static docking port with 6 DOF

This scenario was simulated multiple times under two different conditions. Each time the starting condition was randomized. With a significant adjustment near the target to establish a safe final approach course, as discussed in the challenges section, *Scenario 1(a)*, and with minimal adjustment to create the most optimal route, *Scenario 2(a)*.

The results of each demo scenario show the controllers effectiveness and are shown in the Fig. 3 and Fig. 4 and Table I and Table II. The objective position is set to all zeros for the simplicity of this scenario.

Notice in Fig. 3 how upon entering a specific selectable range the course changed to establish a more extended final approach. This is in comparison to *Scenario 2(b)* where the optimal route is taken directly to the positioning for the final and alignment stages.

TABLE I. SCENARIO 1(A) RESULTS

	Scenario 1(A) Starting and Final Positions					
	X pos	***Y pos***	***Z pos***	α	β	γ
Start	124	70	102	36	7	21
Finish	.04	0	0	.0002	0	0

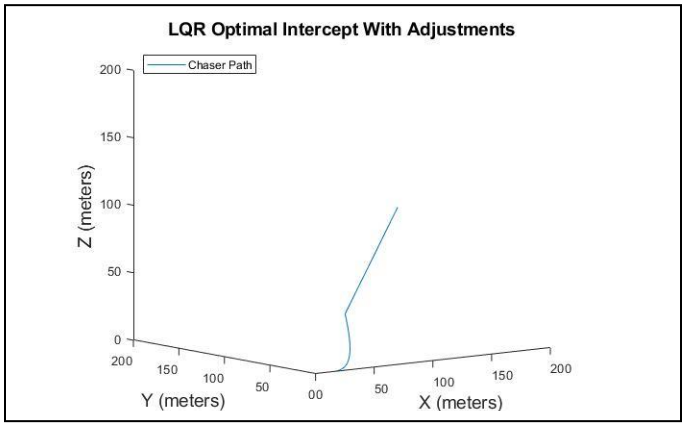

Fig. 3. Scenario 1(a) graphical depiction of LQR controlled course

TABLE II. SCENARIO 1(B) RESULTS

	Scenario 1(A) Starting and Final Positions					
	X pos	***Y pos***	***Z pos***	α	β	γ
Start	160	86	182	16	24	13
Finish	.03	0	0	.0001	0.002	0

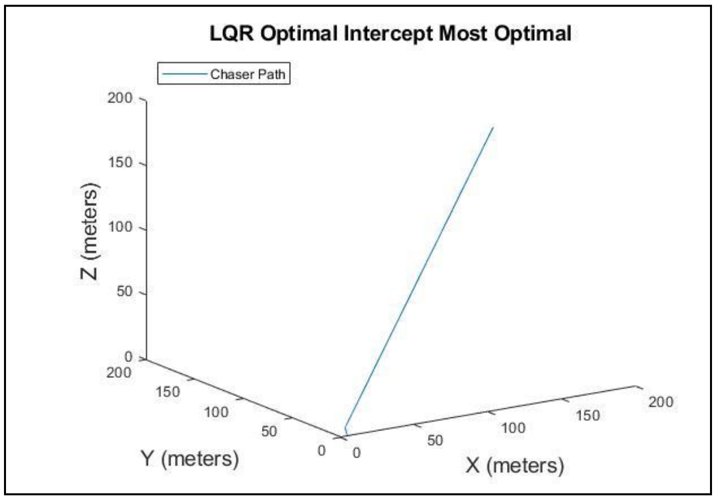

Fig. 4. Scenario 1(b) graphical depiction of LQR controlled course

2) We use an LQR controller to simulate rendevous and docking with a non-rotating but predictably moving docking port with 6 DOF

The rendezvous scenario with no rotation predicted the optimal docking point in Fig. 4 and Table II. The LQR algorithm was able to converge on this position as shown in Fig. 4. By running multiple scenarios similar to *Scenario 2* and analyzing the results of the simulations we determined that the primary factor in finding the minimum intercept point is the closeness of the target on its path to the start position. The alignment is a larger factor when the rotating target is rotating at speeds chaser's rotational limit. In this specific scenario we see that the LQR processor picks nearly the shortest route even though this route still requires small pitch, roll, and yaw alignment.

TABLE III. SCENARIO 2 RESULTS

	Scenario 2(A) Starting and Final Positions					
	X pos	***Y pos***	***Z pos***	α	β	γ
Chaser Start	200	200	200	90	90	90
Intercept Point	19.84	2.53	.08	0	0	0

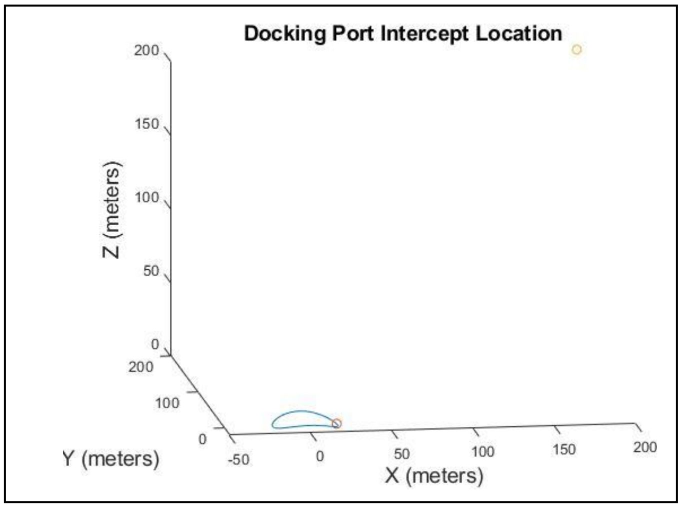

Fig 5. Scenario 2 graphical depiction of LWR optimal intercept point.

Fig. 5.

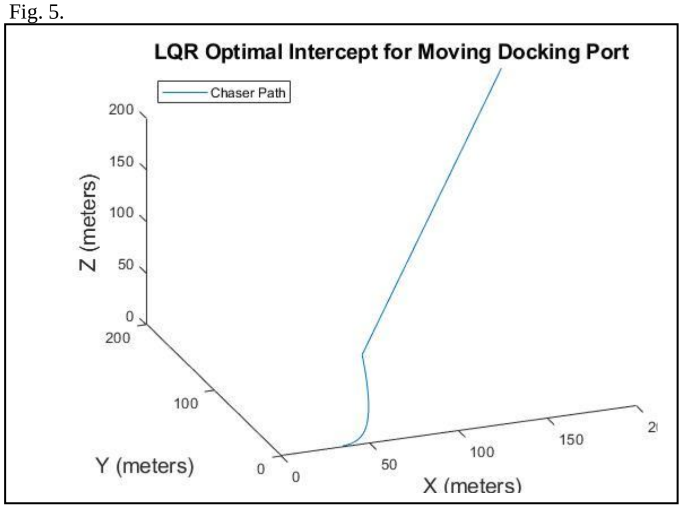

Fig. 6. Scenario 2 graphical depiction of LQR controlled course

3) We use an LQR controller to simulate rendevous and docking with a rotating and predictably moving docking port with 6 DOF

In this scenario the MATLAB program is set such that the target docking port is now rotating in a random direction by up to .26 radians per time step.

Consistent with *Scenario 2* we found that the most significant factor for minimizing the intercept was the closeness to the starting position. Again, the LQR path led to near zero docking position as shown in Fig. 5. The starting position and calculated intercept position are shown in Table IV. Note that the random rotation rate only slightly changed the optimal intercept location.

TABLE IV. SCENARIO 3 RESULTS

	Scenario 3(A) Starting and Final Positions					
	X pos	***Y pos***	***Z pos***	α	β	γ
Chaser Start	200	200	200	90	90	90
Intercept Point	19.35	5.02	.31	1.09	1.24	1.17

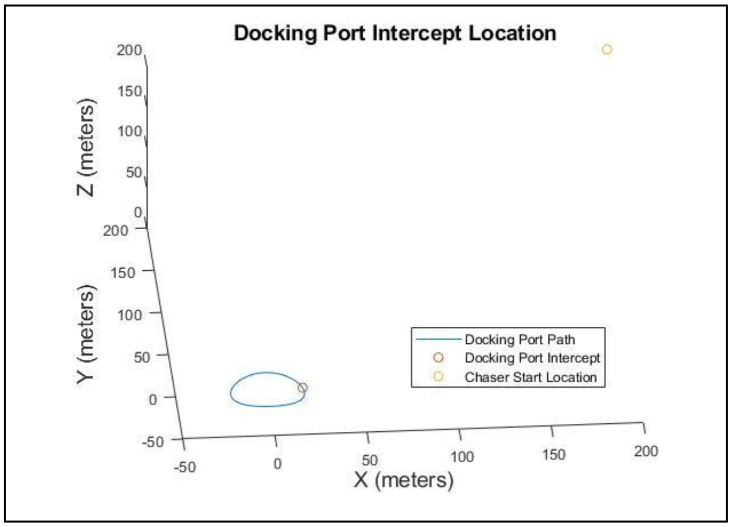

Fig. 7. Scenario 3 graphical depiction of LQR optimal intercpt point

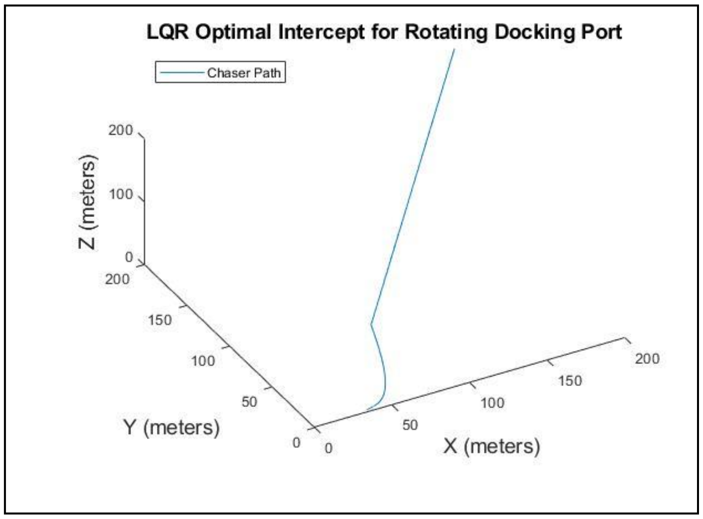

Fig. 8. Scenario 3 graphical depiction of LQR controlled course

B. *Discussion of Computation Time*

Note: The simulations were performed using an Intel i7-7700 @ 3.6 GHz on a desktop computer.

a) Non-Moving Target For the non-moving target scenario where the chaser starts approximately 250 meters from the target the compution of the entire path takes 2.1 seconds. To find each LQR descison or more specifically u^* the computation time was .006 seconds. This is fast enough to rapidly and accurately control the chaser during the rendevous procedure. Additionally this small computation time is fast enough to reroute the entire course without a sgnificant sacrifice in optimality if the chaser had to avoid debris.

b) Moving Target Both moving target scenarios are computationally more expensive. The time to compute the total path from the chaser start position to the docking position was 318 seconds. When peroming calculations at the start position this is not convinient, but accuracy in the computation is still preserved. Additionally, the computation time for each course adjustment calculation was small, only .0015 seconds. Thus the chaser could still use the LQR controller to avoid debris as a safety margin, but afterwards the required time to compute the new optimal path to the target is too long. Thus in the case the chaser must avoid debris or the path of the target changes significantly the chaser can still reach the target, but sacrifices optimality. In *Scenario 2* specifically, where the target completes a recurring path, the cost of calculation time is reduced since the targets path can be predicted.

C. *Comparison to other methods*

Another method for automated orbital docking with a target using sliding mode control is published by Abdollahzadeh and Esmailifar [8]. Simply, this method allows the chaser to calculate and predict the position of the target and then adjust its own attitude and position to directly dock with the target. This method skips the step of positioning the chaser to a point slightly further from the target and accounts for perturbations which may cause collisions.

Comparatively the LQR method we have developed still requires an intermediate point before beginning the final docking procedure. This disadvantages the LQR method for achieving the most optimal procedure. In fact, the LQR controller struggles to calculate the docking procedure on an extremely randomly moving target due to the computation cost on moving targets. This makes the LQR method significantly less effective against truly randomly moving targets, such as intentionally evasive targets. The sliding mode control method is better at accounting for more severe random motion in the target.

The LQR method, however, is still useful compared to sliding mode control. The LQR controller is both cheaper in computational power, less complex, and in the presented scenarios accurately converges to the target vehicle. Both control systems in simple scenarios can accurately close to near zero distances. Thus, the LQR controller may be a better choice in less complex scenarios since it controls accurately in less complex scenarios for less computational cost.

V. CONCLUSION

The project designed and implemented a solution for optimizing the control of a spacecraft docking maneuver. Some of the main challenges involved in this project were the constraints on spacecraft position and angle and ensuring the trajectory does not impact the docking port while still arriving at that location. The initial position of the spacecraft was randomized to evaluate the performance across multiple initial conditions, and several scenarios were examined. The scenarios included a stationary docking port, a docking port moving in a circular trajectory, and a docking port with a complex trajectory. In each of these scenarios, the spacecraft was able to arrive at positions and angles of zero relative to the docking port, within tolerance.

The primary improvements that could be made to the LQR optimal docking control method are reducing the time to compute the optimal docking path to a moving target and adjusting the final docking phase to enable the chaser to dock to an evasive target.

Computation time could be reduced by ignoring predictions about the targets path and simply chasing the target by shifting the LQR target position. This, however, does not solve the evasive target problem which would require the LQR controller to make a smaller set of predictions and if it believed that the target was acting along the path of one of its predictions it could accomplish the docking procedure. Yet, this depends partially on luck and would not be complete optimal control.

Additionally expanding the scope of the project may be useful. For example, investigating and studying non-cooperative docking. Particularly in complex debris fields, where rapid changes in the approach profile are required. This would involve attitude and orbital changes to the chaser vehicle before commencing standard cooperative docking. This has further applications in assembling mission components and capture missions. For now, the near future involves research and modelling to support autonomous non-cooperative capture missions.

Clarification of work

Madson: Wrote and implemented code, wrote challenges and findings section of final report.

Pelster: Helped with code, created presentation, wrote abstract, problem formulation, and conclusion sections of final report.

Srivastava: Searched for references relevant to report. Wrote the problem motivation and literature review section of final report.

Our finalized MATLAB code is available upon request via email: dmadson2@illinois.edu

References

[1] Nandagopal J L, ; Lethakumari R, (2015). [IEEE 2015 International Conference on Technological Advancements in Power and Energy (TAP Energy) - Kollam, India (2015.6.24-2015.6.26)] 2015 International Conference on Technological Advancements in Power and Energy (TAP Energy) - Optimal control of Spacecraft Docking System using integral LQR controller. , (), 18–22. doi:10.1109/tapenergy.2015.7229586

[2] Guarnaccia, Leone; Bevilacqua, Riccardo; Pastorelli, Stefano P. (2016). Suboptimal LQR-based spacecraft full motion control: Theory and experimentation. Acta Astronautica, 122(), 114–136. doi:10.1016/j.actaastro.2016.01.016

[3] Yang, Yaguang (2012). Analytic LQR Design for Spacecraft Control System Based on Quaternion Model. Journal of Aerospace Engineering, 25(3), 448–453. doi:10.1061/(ASCE)AS.1943-5525.0000142

[4] Beatty, Scott (2006). [IEEE 2006 World Automation Congress - Budapest, Hungary (2006.07.24-2006.07.26)] 2006 World Automation Congress - Comparison of PD and LQR Methods for Spacecraft Attitude Control Using Star Trackers. , (), 1–6. doi:10.1109/wac.2006.375957

[5] Starin, S.R.; Yedavalli, R.K.; Sparks, A.G. (2001). [IEEE Proceedings of American Control Conference - Arlington, VA, USA (2001.6.25-2001.6.27)] Proceedings of the 2001 American Control Conference. (Cat. No.01CH37148) - Design of a LQR controller of reduced inputs for multiple spacecraft formation flying. , (), 1327–1332 vol.2. doi:10.1109/ACC.2001.945908

[6] H.Dong, Q. Hu, and M. Akella, “Safety Contol for Spacecradt Autonomous Rendezvous and Docking Under Motion Constraints.” Jnl of Guidance, Control, and Dynamics, vol. 40, pp. 1680–1692, April 2017.

[7] D. Xu, “A Collison-Avoidance Control Algorithm for Spacecraft Proximity Operations Based on imporved Artificial Potential Function.” Jnl of Theoretical and Applied Mechanics, vol. 52, pp. 1581–1589, Dec 2020.

[8] P. Abdollahzadeh and S. Esmailifar “Automatic orbital docking with tumbling target using sliding mode control,” Advances in Space Research, vol 67, pp. 1506-1525, 2021.