Single-Trip Multiple Space Debris Rendezvous: An Integrated APF-MPC Approach

Patra Hsu[1] and Manasvini Srinivasan[1]

Abstract— **Collisions between space debris create more debris, leading to a cascading phenomenon that is a significant safety problem due to the collision risk it poses to satellites and space exploration. Active large debris removal has a high-risk reduction, but it is very costly to implement and there are considerations with the problem of thruster saturation, which notably hampers the maneuverability of the spacecraft used for removal. Previous research explored space navigation using solely Artificial Potential Fields (APF) or APF in conjunction with other local controls, but they fall short of offering a comprehensive solution that combines multi-target global path planning with obstacle avoidance and local control in tandem for effective space navigation. Here, this paper proposes a hybrid Artificial Potential Field - Model Predictive Control (APF-MPC) method that navigates through a known, static space debris field to rendezvous with multiple large target static debris in a singular trip before returning to the start location. The locations of the target debris are queued in order of collection by the method of traversing a polygon-based path. Then, APF is applied between each of the targets, and MPC controls the trajectory of the chaser that navigates through the static obstacles. The MPC strategy provides an enhanced trajectory optimization from the start to the goal location in conjunction with the APF. To simulate our algorithm, we utilize MATLAB for algorithm development and dynamic system modeling. The effectiveness of our approach is assessed using a range of performance metrics, including trajectory smoothness, fuel efficiency, duration of the mission, and efficacy in avoiding obstacles. Our algorithm is benchmarked against established methods such as the A* algorithm and the basic APF approach, allowing us to evaluate the enhancements we have made in terms of mission success rates, fuel economy, and overall operational efficiency. The integration of APF-MPC in our project notably enhances thruster activity planning, leading to more fuel-efficient maneuvers and a marked improvement in computational efficiency.**

I. Introduction

Space debris is an escalating threat to current and future space missions, with over 29,000 tracked pieces larger than 10 cm as of 2023 [1]. Collisions with even small objects at high speeds can catastrophically fragment satellites, exacerbating the growing debris issue [2]. Nearly half of this debris results from satellite disintegration or collisions, amplifying the challenge [2]. Attempts at mitigation strategies like debris removal missions and strict disposal protocols have had limited success, only slightly slowing the debris growth [3]. Unpredictable collisions and intentional acts, like the 2021 Russian anti-satellite test producing 1500 trackable debris pieces, underscore the ongoing risks and vulnerability of our orbital space [4].

This work was not supported by any organization

[1]Patra Hsu and Manasvini Srinivasan is with the Department of Mechanical Engineering, Boston University, 110 Cummington Mall, MA 02215, United States patra@bu.edu and msrini@bu.edu

The Kessler Syndrome exacerbates concerns, where increasing debris triggers further collisions, posing significant threats to satellites and missions [2]. Despite efforts, the complex nature of space and persistent incidents like the 2009 Iridium 33 - Cosmos 2251 collision, as well as intentional tests like China's 2007 anti-satellite missile test, continue to compromise future space access [4].

A. Prior Work

Prior work have explored various approaches to address the growing challenge of space debris removal, including the use of Artificial Potential Fields (APF) for obstacle avoidance and Model Predictive Control (MPC) into space navigation strategies.

1) Artificial Potential Fields in Space Navigation: APF for obstacle avoidance was implemented for spacecraft guidance and control in [5]. Here, they present a methodology that allows for real-time control of complex maneuvers with minimal computational power and avoidance of obstacles. APF is excellent for static obstacle avoidance, but it does not adapt well to moving obstacles [6]. In addition, APF is prone to traps in local minima in cluttered environments, such as space debris fields. Therefore, there is a need to add a condition to the APF for local control and trajectory optimization with dynamic obstacles in addition to the global path planning APF provides. [7] introduces a decentralized autonomous low-thrust control approach to address the challenge of efficiently gathering micro-satellite clusters with APF for autonomous operation. The paper falls short in terms of optimizing the trajectory for cluster convergence, relying solely on APF, which is inefficient in low-thrust applications. Here [8], autonomous collision avoidance strategies are used for the rendezvous of spacecrafts implementing an enhanced potential function. The paper utilizes methods for autonomously guiding spacecrafts towards large debris targets while avoiding obstacles along the trajectory. However, while it considers both dynamic and static obstacles, it solely relies on APF for the development of an optimal path. Furthermore, it incorporates the constraints of thrust saturation as part of the problem statement. Our approach employs a polygon method that allows for the collection of multiple debris targets opposed to singular targets. Additionally, we use MPC to enhance path planning and effectively address thrust saturation issues.

2) Artificial Potential Fields and Controls: [9] Enhances the APF method for optimal local control and trajectory,

using a unified APF and parallel navigation algorithm. While parallel navigation adjusts the robot's path to avoid obstacles while following a reference route, this study focuses on modeling static obstacles. Nevertheless, it achieved a notable 14.4% to 21.4% reduction in time to reach the goal compared to APF, suggesting the potential benefits of fine-tuning APF with local control and trajectory optimization. This paper validates the addition of local control to APF, and we build upon this validation by looking into adding the best local control to APF for our environment and use case. [10] Applies the APF method alongside sliding mode techniques (SMC) to a orbital simulator with obstacles. This paper also deals with static obstacles and it improves the success rate of rendezvous simulations with obstacles compared to using just APF. The paper acknowledges a limitation of SMC, known for chattering, which pose challenges in space due to thruster wear and tear. We build upon this paper by veering away from SMC methods due chattering challenges.

3) Model Predictive Control: Model Predictive Control significantly improves the trajectory of a spacecraft as seen here [11]. The paper introduces the utilization of MPC, through MATMPC, as a control method for the spacecraft's maneuvers. However, this proposal does not delve into the crucial aspect of obstacle avoidance. Our approach introduces the integration of artificial potential methods into the spacecraft's control . In both these papers [12], [13] emphasis is placed on addressing obstacle avoidance for moving and static obstacles, employing MPC techniques to calculate an optimal trajectory for the spacecraft. But its focus is on obstacle avoidance, without methods to guide the spacecraft to its destination. We combine the use of APF both for reaching the goal and obstacle avoidance. We leverage MPC to improve trajectory and avoid obstacles effectively.

4) Model Predictive Control and Artificial Potential Fields: [14] combines APF with MPC for trajectory tracking and obstacle avoidance for mobile robots. APF constructs a virtual goal with a vision-based sensor, followed by MPC to optimize control inputs to track the reference trajectory. MPC for local control can be robust to cluttered dynamic environments; however, this paper utilizes a vision sensor to map a virtual goal when the obstacle is detected, whereas in spaceflight it may be difficult to use vision-based sensors. Similarly, [15] utilizes APF and MPC for obstacle avoidance but in midair where multiple robots are navigating to a goal while avoiding collision and obstacles. Results show the reference paths generated previously using the APF path planner coincide with the actual paths moved by the MPC controller, showing feasibility of accurate local control and path planning. However, we build upon this paper by implementing a similar models but with a single, autonomous chaser and multi-target objectives.

B. Paper Contribution

Our implementation of this paper involves the generation of an simple trajectory aimed at reaching the target debris by utilizing global APF and local control MPC method and justifying it with other path planners. This proposed approach is designed to optimize the trajectory for long-duration space maneuvers, often spanning several days. By focusing on the calculation of precise reference trajectories, our method aids in significantly enhancing fuel efficiency. This strategic approach ensures the trajectory remains feasible and efficient, effectively addressing the challenges of extended space missions.

II. Methodology

A. Multi-Target Debris Queue Ordering

In pursuit of efficient navigation to reach the target debris, the methodical planning of the spacecraft's trajectory is important for the successful rendezvous while effectively managing available resources. To accomplish this objective, the development and utilization of a meticulous queuing system are instrumental. This system serves as a fundamental component, aimed at developing an improved path for the spacecraft, thereby curbing fuel consumption while ensuring computational efficiency. The Multi-Target Debris Queue Ordering detailed below encompasses a strategic approach that implements the ordering and traversal of multiple target debris points, creating a streamlined pathway for the spacecraft's safe navigation amidst the challenging space terrain.

A polygonal path generation strategy is employed to achieve this systematically arranging the sequence and traversal of multiple target debris points.

B. Artificial Potential Field Method

Artificial Potential Field Path Planning is a motion planning technique that models the environment as a potential field, with attractive forces guiding the robot towards the goal and repulsive forces steering it away from obstacles [7]. The robot navigates through the field by moving along the steepest descent of the potential function, resulting in a smooth trajectory. We utilize this approach for its simplicity and effectiveness in generating collision-free paths for our spaceship with limited on-board computing. Once we have the queue of the target points for retrieval, the APF path planning can be implemented between each of the targets to avoid collision with other debris and to reach the next target.

C. Non-linear Model Predictive Control.

Non-linear Model Predictive Control (MPC) is employed in our implementation as a sophisticated control strategy, specifically designed to enhance trajectory optimization. This approach involves minimizing a predefined cost function, while adhering to the system's constraints over a finite future time frame. We actively analyze and predict the current state of our space navigation system through a series of complex control actions. These actions are derived by solving a non-linear constrained optimization problem, with the goal of reducing the total cost over a pre-set horizon.

In this non-linear framework, MPC only implements the initial control action from the calculated sequence at each step, while the subsequent actions are recalculated in the

next iteration. This iterative process is crucial for adaptively modifying control actions in response to the evolving state of the system, thereby accommodating the non-linear dynamics of space navigation.

By integrating non-linear MPC, we significantly refine the trajectory initially derived from the Artificial Potential Field (APF) method. This integration allows for real-time adjustments to control inputs, adeptly responding to the changing spatial conditions. Such an approach not only enables adaptive trajectory modification but also aids in reducing overall operational costs throughout the mission's duration. This aligns seamlessly with our project's objectives of achieving high efficiency and precision in the challenging task of space debris removal.

III. Algorithm Implementation

A. Multi-Target Debris Queue Ordering

After identifying the locations of the large target debris the spacecraft will rendezvous with, we first need to find an optimal queuing method for the targets to be collected. The goal of this queue ordering is to minimize the fuel and the on-board computational consumption of the spacecraft whilst maximizing targets retrieved in a single trip [16].

This algorithm utilizes the concept of polygonal path generation that is based loosely on the Hamilton circuit algorithm [17] from randomized 2D points in a constrained field to achieve the optimal multi-target retrieval queue while visiting each target once. However, unlike the Hamiltonian circuit, this path generation also prevents trajectory paths from ever crossing each other.

To do so, the algorithm first finds the min and max value along the x-axis from the target points in the vector targetPoints and the (x,y) index of those 2 points (minXTarget and maxXTarget). Then, it computes a line between the points, finding the slope and y-intercept. The line between the two points is then utilized to divide the rest of the points into 2 vectors of targetPoints, one above the line (aboveTargetPoints) and one below the line (belowTargetPoints). The elements in aboveTargetPoints are then sorted by column by ascending x values, and concurrently, the elements in belowTargetPoints are sorted by descending x values. Then, the two vectors are concatenated together, and the first element of aboveTarget-Points is appended to the end of the vector, giving the vector returned from this algorithm called totalSortedTargets. This algorithm results in an order that a spacecraft (chaser) can traverse through to meet with all of the targeted points and return back to the same base it started off in the beginning from.

The result of this is a robust polygonal queue algorithm that always orders the points so that a closed-loop polygonal path can be drawn without any of the path edges crossing each other. Although the purpose of the implementation assumes the target debris locations are already defined, this algorithm was generalized for randomized target locations and number of targets to show robustness of the algorithm and to test for edges cases.

Algorithm 1 Multi-Target Debris Queue Ordering Algorithm

Input: NTargets, N
Output: Order-Sorted Targets, T, of size (2 x N)
Initialization
  $t(x, y), \ldots$ Randomize Points **for** $i = 1$ *to* N **do**
    $t \leftarrow 10 \times \text{rand}([2, N(i)])$
  end
end
Find min and max value in the x axis.
Find indexes in y axis.
Extract MinX and MaxX.
Calculate line between $t(\text{MinX})$ and $t(\text{Max})$.
$Slope \leftarrow \frac{y_2 - y_1}{x_2 - x_1}$
$Y\text{-}intercept \leftarrow y_1 - \text{Slope} \times x_1$
Sort Targets based on if it is above or below the line
for $i = 1$ *to* N **do**
  if $t(2, i) > Slope \times t(1, i) + Y\text{-}intercept$ **then**
    $aboveTargets \leftarrow t(:, i)$
  end
  else
    $belowTargets \leftarrow t(:, i)$
  end
end
Sorting targets:
$aboveTargets \leftarrow$ sort columns by ascending x value
$belowTargets \leftarrow$ sort columns by descending x value
$T \leftarrow$ Combine$[aboveTargets, belowTargets]$
Append$[T, T(:, 1)]$
return T

Fig. 1: Visual of polygonal path drawn by the Multi-Target Debris Queue Ordering Algorithm. The pink box denotes the docking station where the path starts and ends on.

B. APF Global Trajectory Path Planning

Here, we overlaid the targets on top of a predefined world where spheres are denoted as obstacles to represent non-targeted space debris. As a proof of concept in a 2D world,

we are under the assumption that targets and obstacles do not overlap. Hence, a seed was set to lock the randomized target points in the Multi-Target Debris Queue Ordering algorithm outputs, and obstacles defined in a struct *world* are manually set between the targets to test the AFP-based path planner and saved into a .mat file *sphereWorld1.mat*. The goal vector *xGoal* was created from vector *totalSortedTargets*, where it takes all of the targets except for the first one, and correspondingly, vector *xStart*, where it takes all of the targets except for the last one. Each x_{Goal} is then defined with the attractive potential of:

$$dP(x, x_{\text{goal}}) = U_{\text{attractive}}(x) = \|x - x_{\text{goal}}\|^2 \tag{1}$$

Where:

$p = 2$ (quadratic potential),
$dP(x, x_{\text{goal}})$ (attractive potential function),
$U_{\text{attractive}}(x)$ (attractive potential at position x),
$\|x - x_{\text{goal}}\|$ (squared Euclidean distance)

Then the gradient of the attractive potential function $U_{\text{attractive}}(x)$ is given by:

$$\nabla U_{\text{attractive}}(x) = p \cdot d_p^{p-2}(x, x_{\text{goal}}) \cdot (x - x_{\text{goal}}) \tag{2}$$

The obstacles are then assigned with the repulsive potential of:

$$U_{\text{rep},i}(x) = \begin{cases} \frac{1}{2}\left(\frac{1}{d_i(x)} - \frac{1}{d_{\text{influence}^2}}\right) & \text{if A,} \\ 0 & \text{if B,} \\ undefined & \text{if C} \end{cases} \tag{3}$$

With the corresponding gradient of :

$$\nabla U_{\text{rep},i}(x) = \begin{cases} -\left(\frac{1}{d_i(x)} - \frac{1}{d_{\text{influence}}}\right)\frac{1}{d_i(x)^2}\nabla d_i(x) & \text{if A,} \\ 0 & \text{if B,} \\ \text{undefined} & \text{if C} \end{cases} \tag{4}$$

Where:

case A if $0 < d_i(x) < d_{\text{influence}}$,
case B if $d_i(x) > d_{\text{influence}}$,
case C otherwise.

Then, the total potential and the gradient of the total potential function are calculated as follows, with α being a repulsion factor.

$$U = U_{\text{attr}} + \alpha \sum_i U_{\text{rep},i} \tag{5}$$

$$\nabla U = \nabla U_{\text{attr}} + \alpha \sum_i \nabla U_{\text{rep},i} \tag{6}$$

Maximum steps NSteps = 400 and step size epsilon = .01 are then defined, and for each step, the cost and location of the planner is calculated and returned, and the spacecraft moves in the direction of the negative total gradient. Once the spacecraft reaches the *xGoal*, the *xGoal* becomes the *xStart*, and the next queued target becomes the new *xGoal*. This iterates until the planner reaches the end of the *totalSortedTargets* vector, or the planner reaches the step limit first.

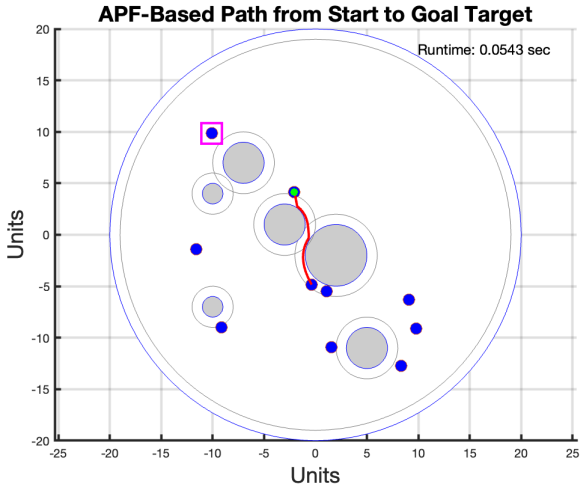

Fig. 2: The second segment of the path planner avoiding debris obstacles and reaching the third target. The green dot is the starting location of the segment, and the pink square denotes the starting dock. The local runtime is 0.0543 seconds. The distance of influence is 1.

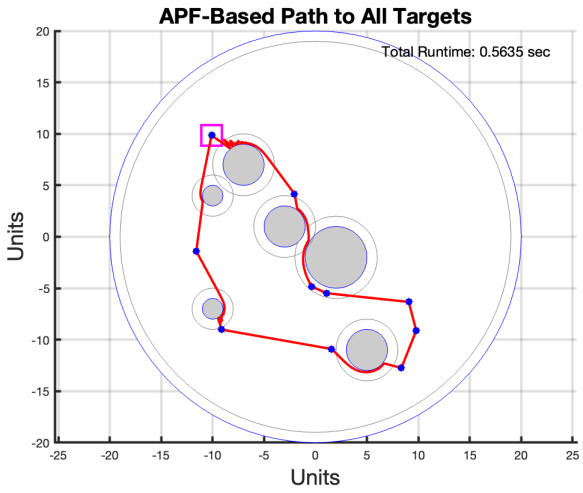

Fig. 3: The full path of the path planner (red path) visiting all targets (blue dots) in the order described in Fig.1. And returning to the home dock. The full runtime is .5635 seconds.

IV. Non-linear Model Predictive Control (MPC)

The Non-linear Model Predictive Control (MPC) algorithm is integrated into a simulated 2D space environment, where

predefined trajectories are refined to navigate through a field of potential targets and obstacles. This field is represented by the `totalSortedTargets` vector, which lists key waypoints for the spacecraft to reach. The MPC algorithm optimizes the trajectory between these waypoints, considering both the efficiency and dynamics of space navigation.

The MPC implementation employs a non-linear kinematic model (`kinematicDynamics2D`) to predict the spacecraft's future state based on its current position, velocity, and control inputs (acceleration). The control actions are optimized through a non-linear optimization process (`fminunc`), which iteratively refines the control inputs to minimize a custom cost function. This function encompasses factors like tracking error, control effort, and trajectory smoothness, balancing the trade-off between accuracy and resource utilization.

Key parameters in the MPC process include:

- Prediction Horizon $N = 20$, determining the lookahead steps in trajectory planning.
- Time step $dt = 0.1$, defining the granularity of control action adjustments.
- Weights for control effort (`ctrlEffortWeight`) and trajectory smoothness (`smoothnessWeight`), allowing fine-tuning of the optimization process.

The refined trajectory (`refinedTraj`) is computed in segments between each pair of subsequent target points. For each segment, the MPC algorithm calculates the optimal series of control actions that guide the spacecraft from its current state to the next target. The first control action of the optimized sequence is applied, and the process is repeated for the next segment, ensuring that the trajectory adapts to the evolving spatial scenario.

Metrics like path length, fuel consumption, smoothness, and time to completion are calculated (`calcTrajMetrics`) to evaluate the performance of the refined trajectory. These metrics offer a quantitative assessment of the improvements brought by the MPC in terms of operational efficiency and mission success.

Overall, our Non-linear MPC algorithm plays a pivotal role in our space navigation system, enabling adaptive, efficient, and precise trajectory planning amidst a dynamically simulated space environment. This equation represents the kinematic dynamics of the spacecraft in 2D space, considering position, velocity, and acceleration.

$$\begin{aligned} new\_x &= x + vx \cdot dt + \frac{1}{2} \cdot ax \cdot dt^2, \\ new\_y &= y + vy \cdot dt + \frac{1}{2} \cdot ay \cdot dt^2, \\ new\_vx &= vx + ax \cdot dt, \\ new\_vy &= vy + ay \cdot dt. \end{aligned} \tag{7}$$

The cost function used in the MPC optimization process, considering tracking error, control effort, and smoothness.

$$\begin{aligned} totalCost = \sum_{j=1}^{N} \big[&\|xTemp_j(1:2) - x_{\text{target}}\|^2 \\ &+ \lambda_1 \cdot \|u_j\|^2 + \lambda_2 \cdot \|\Delta u_j\|^2 \big], \end{aligned} \tag{8}$$

where λ_1 and λ_2 are the weights for control effort and smoothness, respectively, and Δu_j represents the change in control input between consecutive steps.

This equation encapsulates the non-linear optimization problem solved at each step in the MPC.

$$\begin{aligned} \min_{u} \quad & totalCost(u, x_{\text{initial}}, x_{\text{target}}, dt, N, \lambda_1, \lambda_2) \\ \text{subject to} \quad & x_{\text{initial}},\ x_{\text{target}},\ u \in \mathcal{U}, \end{aligned} \tag{9}$$

where $\mathcal{U}$ represents the set of allowable control actions.

This equation represents the iterative process of trajectory refinement in the MPC.

$$\begin{aligned} \text{refinedSegment}_{j+1} = \text{kinematicDynamics2D}(\text{refinedSegment}_j dt), \\ j = 1, 2, \ldots, N, \end{aligned} \tag{10}$$

where $u_{\text{optimized}}$ is the optimized control input from the MPC.

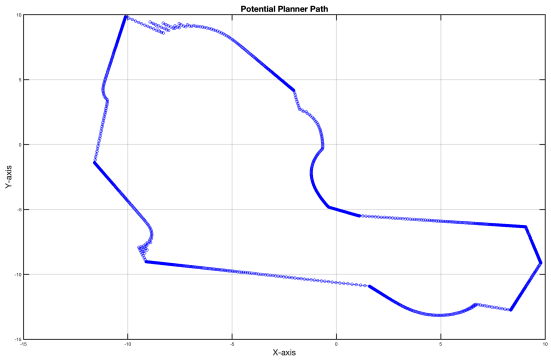

Fig. 4: APF-based trajectory: The blue trajectory represents the full path generated by the Potential Planner. The obstacles and targets are removed to show the refinement of the trajectory generated by the APF-MPC approach.

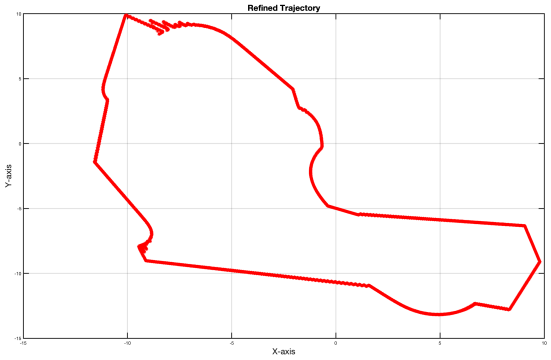

Fig. 5: APF-MPC Trajectory: The red trajectory represents the full path generated by the Integrated APF-MPC algorithm. The runtime for this trajectory is 8,379 seconds. Integrated APF-MPC showcases its capability with a significantly larger number of waypoints and lower maximal acceleration and deceleration, which contribute to a smoother trajectory.

Algorithm 2 Non-linear Model Predictive Control for Trajectory Refinement

Input: xPath; filePath; dt; N; ctrlEffortWeight; smoothnessWeight, S

Output: refinedTrajectory, R

```
xPath ← readmatrix(filePath);
totalSortedTargets ← generateTotalSortedTargets(NTargets;)
refinedTraj ← refineTrajNL(xPath, dt, N, ctrlEffortWeight, S);
refinedTrajSmooth ← smoothTraj(R, windowSize;)
plotTrajectories(xPath, refinedTrajSmooth);
initialMetrics ← calcTrajMetrics(xPath, dt);
refinedMetrics ← calcTrajMetrics(refinedTrajSmooth, dt);
dispMetrics(initialMetrics, refinedMetrics);

Function refineTrajNL(xPath, dt, N, ctrlEffortWeight, S):
    totalPoints ← size(xPath, 2); R ← zeros((N + 1) × (totalPoints - 1), 2); initialGuess ← repmat([0.1, 0], N, 1); currentIdx ← 1;
    for i ← 1 to totalPoints - 1 do
        xInit ← [xPath(1, i); xPath(2, i); 0; 0];
        xTarget ← xPath(:, i+1);
        optimizedCtrls ← fminunc(costFxn, initGuess, options);
        refinedSegment ← computeSegment(xInit, optimizedCtrls, N, dt);
        R(currentIdx:endIdx, :) ← refinedSegment;
        currentIdx ← endIdx + 1;
        initGuess ← optimizedCtrls;
    return R
Function computeSegment(xInit, optimizedCtrls, N, dt);:
    for j ← 1 to N do
        ctrl ← optimizedCtrls(2 × j - 1 : 2 × j);
        xInit ← kinematicDynamics2D(xInit, ctrl, dt);
        refinedSegment(j + 1, :) ← xInit(1:2);
    return refinedSegment;
```

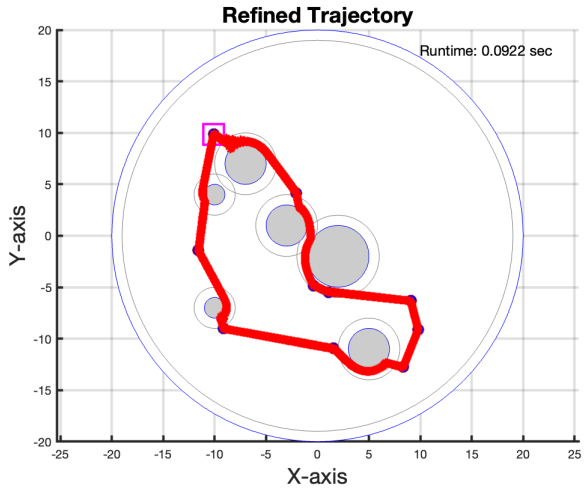

Fig. 6: APF-MPC Trajectory: The red trajectory represents the full path generated by the Integrated APF-MPC algorithm. The runtime for this trajectory is 8,379 seconds. Integrated APF-MPC showcases its capability with a significantly larger number of waypoints and lower maximal acceleration and deceleration, which contribute to a smoother trajectory.

V. Simulation and Experimental Results

A. Integrated APF-MPC

In this section, we present the simulation and experimental results for our integrated Approach Potential Fields (APF) and Model Predictive Control (MPC) approach. The initial trajectory was obtained from a pre-defined path, and our refinement process was applied to optimize the trajectory based on the specified parameters. The refinement process involved solving an optimization problem that minimized a cost function, taking into account control effort and smoothness while tracking the desired path. The results demonstrate a significant improvement in various trajectory metrics, including path length, fuel consumption, smoothness, total time to completion, maximal acceleration, maximal deceleration, number of waypoints, and total curvature. The initial trajectory exhibited characteristics such as high length and significant control effort, resulting in poor fuel efficiency and suboptimal smoothness. However, after the application of the MPC-based refinement, the trajectory was markedly improved. The length of the trajectory was reduced substantially, leading to enhanced fuel efficiency, and the smoothness was significantly increased, indicating a more stable and controlled motion profile. Additionally, the total time to completion, maximal acceleration, and maximal deceleration were adjusted to more desirable values. The number of waypoints also increased, indicating a finer-grained trajectory. Finally, the total curvature of the trajectory demonstrated a substantial reduction, signifying smoother and less abrupt changes in direction.

B. Verification and Validation

To measure the level of success we realized with our Integrated APF-MPC algorithm, we compared it to other path

planners to accurately benchmark it. The baseline potential planner (applied in Section 4b) and the A* algorithm was chosen as alternative models. Both algorithms utilize the same obstacle space and order of targets provided from III.a to provide a controlled environment for testing.

1) A and Baseline Potential Planner Comparison:* The A* algorithm was adapted from Homework 4. The grid size was 20×20, which matched the environment of the path planner. However, the cell (step) size minimum had to be 50 to allow for every obstacle to be surrounded by at least 1 pathway. It is important to note that with this cell size, some of the grids slightly crossed over an obstacle.

Algorithm 3 Simplified A* Algorithm

Output: Runtime of Segments, R

Initialization
  $n_{\text{start}} = O$; $g(n_{\text{start}}) = 0$; $backpointer = 0$; $C = []$
repeat
  Pick n_{best} from O s.t. $f(n_{\text{best}}) \leq f(n_O)$ for all $n_O \in O$
  Remove n_{best} from O and add it to C
  if $n_{best} = n_{goal}$ **then**
    Return path using backpointers from n_{goal}
    end if
  $S = \text{ExpandList}(n_{\text{best}})$
  for $x \in S$ *that is not in* C **do**
    if $x \notin O$ **then**
      Set the backpointer cost $g(x) = g(n_{\text{best}}) + c(n_{\text{best}}, x)$. Set the backpointer of x to n_{best}. Compute the heuristic $h(x)$. Add x to O with value $f(x) = g(x) + h(x)$.
    else
      if $g(n_{best}) + c(n_{best}, x) < g(x)$ **then**
        Update the backpointer cost $g(x) = g(n_{\text{best}}) + c(n_{\text{best}}, x)$
      Update the backpointer of x to n_{best}
until *O is empty*;

Overall, the A* algorithm was able to find paths to each target and was around 0.4 seconds faster than the baseline potential planner algorithm and 8379 seconds faster than the APF-MPC planner.

Although A* and the baseline potential planner algorithm ended up having a faster runtime to compute the paths, the time to run APF-MPC was a realistic representation of the computational cost when compared to other path planning methods commonly utilized, given the constraints and dynamics of the vehicle in the MPC implementation. The usage of the potential planner as a global trajectory planner is ideal over A* as it places more emphasis on obstacle avoidance. Due to the high cost of the actual implementation on hardware in space, it is imperative that the global algorithm chosen has the lowest chances of unplanned obstacle contact [7]. In addition, the potential planner can be executed without having a known environment or known obstacle shape. Depending on grid size, A* may struggle with unnatural, unknown obstacle shapes, such as a piece of debris in a debris field in space [18]. In essence, our test environment employs spheres as obstacles, which may lead to faster A* completion time, as shown in this paper.

Compared to the base potential planner, the APF-MPC was able to achieve significantly improved path length and fuel consumption metrics. Specifically, the APF-MPC algorithm generated a much shorter path with reduced fuel consumption, making it a more efficient choice for trajectory planning. This efficiency is attributed to the MPC's ability to account for both global path planning and local control optimization.

One of the key advantages of the APF-MPC approach is its capability to navigate complex environments effectively. It can better handle situations where the robot encounters tight spaces or obstacles close to the planned path. For instance, as seen on Fig. 3, the first segment has disturbances as it avoids the obstacles and navigates from the dock to the first goal. The local control provided by MPC allows the system to adapt dynamically, avoiding local minima and ensuring smoother and more efficient trajectories.

Furthermore, APF-MPC offers robustness in handling uncertainties in the environment. It can adjust its trajectory in real-time based on sensor inputs, making it well-suited for scenarios where the environment is not perfectly known or changes over time. This adaptability is a critical feature for applications in dynamic and unpredictable environments.

Overall, the APF-MPC algorithm outperforms the base potential planner by providing more efficient and adaptable path planning solutions, making it a promising choice for various robotic applications where trajectory optimization is essential

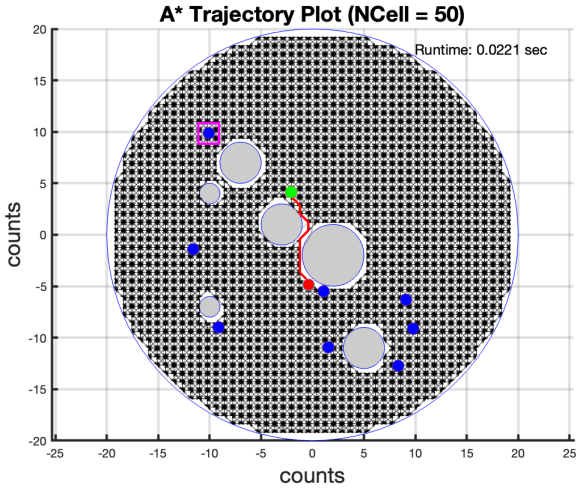

Fig. 7: A* path planner (red path) for one section where the green dot denotes the starting location and the red dot is the goal location. The blue dots denotes the other target locations. The runtime is .0642seconds.

We evaluated the performance of the APF-MPC algorithm with a baseline potential planner using the following metrics:

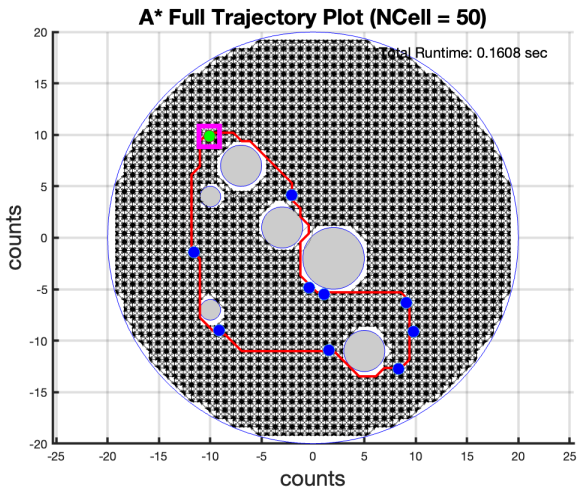

Fig. 8: The full path of the path planner (red path) visiting all targets (blue dots) in the order described in Fig.1. And returning to the home dock. The full runtime is .1341 seconds.

- **Path Length (meters):** The length of the generated trajectory.
- **Fuel Consumption (units of fuel):** The amount of fuel consumed during the trajectory.
- **Smoothness:** A measure of trajectory smoothness.
- **Total Time (seconds):** The time taken to complete the trajectory.
- **Maximal Acceleration (m/s^2):** The maximum acceleration experienced along the trajectory.
- **Maximal Deceleration (m/s^2):** The maximum deceleration experienced along the trajectory.
- **Number of Waypoints (count):** The total number of waypoints in the trajectory.
- **Total Curvature (radians):** The sum of all angle changes in the trajectory.

The following table summarizes the key metrics obtained from these comparisons, highlighting the superior performance of our Integrated APF-MPC approach:

TABLE I: Comparison of Path Planning Algorithms

Metric	Potential Planner	Integrated APF-MPC
Path Length	841.7818	105.1884
Fuel Consumption	2367.9007	69.5320
Smoothness	3.6116	3.8684×10^{-4}
Total Time to Completion	0.2000	8379
Maximal Acceleration	361.1585	21.8227
Maximal Deceleration	361.1585	2.2462×10^{-7}
Number of Waypoints	2	83790
Total Curvature	0	2.3422×10^{4}

VI. Conclusion

In this paper, a series of algorithms were described and implemented to create a novel, integrated APF-MPC proof-of-concept approach that traverses through a static obstacle field to rendezvous with multiple static targets in one trip and returns to the dock. From the authors' best knowledge, there have not been any work on single-trip multiple space debris collection by the completion of this paper. By this method, a global potential path planner is improved upon with the addition of local control with MPC, allowing for path planning to also take into account critical factors such as trajectory optimization and fuel consumption. This approach is especially crucial in space environments, where the prioritization of trajectory precision, fuel efficiency, and minimization of hardware implementation cost are of paramount importance. The key strengths and contributions of our approach include:

1) **Efficient Path Planning:** Our algorithm has demonstrated exceptional performance in terms of path length and fuel consumption. The use of MPC allows us to generate shorter and more efficient paths, which are essential for missions with limited fuel resources.
2) **Adaptive Obstacle Avoidance:** The APF-MPC algorithm excels in navigating through complex environments with obstacles. Its local control capability enables dynamic adjustments to avoid local minima and ensures smoother trajectories.
3) **Robustness to Uncertainty:** The adaptability of our approach is particularly valuable in scenarios where the environment is not perfectly known or may change unpredictably. This feature is crucial for addressing real-world challenges in space debris collection.
4) **Cost-Effective Implementation:** By minimizing the cost of hardware implementation, our approach aligns with the high-priority objectives of cost-efficiency and safety in space missions.

Comparative tests against A* and the base potential planner highlight the superiority of our APF-MPC algorithm. While A* and the baseline potential planner showed commendable performance, our approach consistently outperformed them, especially in terms of efficiency and adaptability.

VII. Future Work

This novel algorithm implementation shows a successful implementation proof-of-concept solution to multi-target rendezvous in space. Due to the robustness of the code to allow for easy modification to include control functions that would simulate the true dynamic of different space environments, we postulate that future works would be able to easily build on top of the described work here. Future research in this domain can encompass the implementation of dynamic and static obstacle avoidance mechanisms using MPC. The MPC framework can be extended to address dynamic obstacles that may enter the path of the spacecraft during the mission. Real-time sensing and decision-making capabilities can be integrated to adapt the trajectory and ensure collision-free navigation. Furthermore, incorporating static obstacle avoidance with MPC is essential for accounting for known obstacles such as satellites, space stations, or celestial bodies. Simulation of gravitational pull and other orbital dynamics would also need to be considered. Lastly, the queuing of targets can be more optimized for the shortest path

in off-line computing—in essence—calculating the shortest polygonal total edge length.

REFERENCES

[1] Loïs Miraux. Environmental limits to the space sector's growth. *Science of The Total Environment*, 806:150862, 2022.

[2] V.V. Adushkin, O. Yu. Aksenov, S.S. Veniaminov, S.I. Kozlov, and V.V. Tyurenkova. The small orbital debris population and its impact on space activities and ecological safety. *Acta Astronautica*, 176:591–597, 2020.

[3] Christophe Bonnal, Jean-Marc Ruault, and Marie-Christine Desjean. Active debris removal: Recent progress and current trends. *Acta Astronautica*, 85:51–60, 2013.

[4] Bohumil Dobos and Jakub Prazak. To clear or to eliminate? active debris removal systems as antisatellite weapons. *Space Policy*, 47:217–223, 2019.

[5] Ismael Lopez and Colin R Mclnnes. Autonomous rendezvous using artificial potential function guidance. *J. Guid. Control Dyn.*, 18(2):237–241, March 1995.

[6] Zheng Huang, Yun Lu, Hao Wen, and Dongping Jin. Ground-based experiment of capturing space debris based on artificial potential field. *Acta Astronautica*, 152, 08 2018.

[7] Yan Yu, Chengfei Yue, Huayi Li, and Feng Wang. Autonomous low-thrust control of long-distance satellite clusters using artificial potential function. *The Journal of the Astronautical Sciences*, 68, 03 2021.

[8] Chuangge Wang, Danhe Chen, Wenhe Liao, and Zhenhua Liang. Autonomous obstacle avoidance strategies in the mission of large space debris removal using potential function. *Advances in Space Research*, 04 2022.

[9] Patrick Friudenberg and Scott Koziol. Mobile robot rendezvous using potential fields combined with parallel navigation. *IEEE Access*, 6:16948–16957, 2018.

[10] Nicoletta Bloise, Elisa Capello, Matteo Dentis, and Elisabetta Punta. Obstacle avoidance with potential field applied to a rendezvous maneuver. *Appl. Sci. (Basel)*, 7(10):1042, October 2017.

[11] Alexander Korsfeldt Larsén. Non-linear model predictive control for space debris removal missions. 2018.

[12] Hyeongjun Park, Stefano Di Cairano, and Ilya Kolmanovsky. Model predictive control for spacecraft rendezvous and docking with a rotating/tumbling platform and for debris avoidance. In *Proceedings of the 2011 American Control Conference*, pages 1922–1927, 2011.

[13] Christopher Jewison, Richard Erwin, and Alvar Saenz-Otero. Model predictive control with ellipsoid obstacle constraints for spacecraft rendezvous. *IFAC-PapersOnLine*, 48:257–262, 12 2015.

[14] Jianhua Li, Jianfeng Sun, Liqun Liu, and Jiasheng Xu. Model predictive control for the tracking of autonomous mobile robot combined with a local path planning. *Meas. Control*, 54(9-10):1319–1325, November 2021.

[15] Rawan Ahmad and Ayman El-Badawy. Artificial potential field for dynamic obstacle avoidance with mpc-based trajectory tracking for multiple quadrotors. 10 2020.

[16] Vladimir S. Aslanov and Alexander S. Ledkov. Fuel costs estimation for ion beam assisted space debris removal mission with and without attitude control. *Acta Astronautica*, 187:123–132, 2021.

[17] Zhong Yu, Liang Jinhai, Gu Guochang, Zhang Rubo, and Yang Haiyan. An implementation of evolutionary computation for path planning of cooperative mobile robots. In *Proceedings of the 4th World Congress on Intelligent Control and Automation (Cat. No.02EX527)*, volume 3, pages 1798–1802 vol.3, 2002.

[18] Daniel Foead, Alifio Ghifari, Marchel Budi Kusuma, Novita Hanafiah, and Eric Gunawan. A systematic literature review of a* pathfinding. *Procedia Computer Science*, 179:507–514, 2021. 5th International Conference on Computer Science and Computational Intelligence 2020.