
# Project Report

## Multiple Target Tracking using Nearest Neighbor method and Game Theoretic Approach

**Prepared by:**

**Polamuri Mohana Krishna**

**B. tech (6th semester)**

**J.K Institute of Applied Physics and Technology**

**University of Allahabad.**

**Guided by:**

**Prof. Dr. Arun Mishra**

**Dept of Computer Science & Engineering**

**Defence Institute of Advanced Technology.**

# Acknowledgement

During the construction of my project, I had to take the help and guidelines of some respected persons, who deserve my utmost and sincere gratitude. The completion of this project gives me much pleasure. I would like to show my gratitude to Prof. Dr. Arun Mishra, Defence Institute of Advanced Technology for giving me a good counsel and guidance for my project throughout its numerous consultations. I would also like to expand my deepest gratitude to all those who have directly and indirectly guided me in making this project a success.

In addition, to everything Professor Dr. Arun Mishra, who introduced me to the Methodology of work, and whose passion for the "underlying structures" had an everlasting effect on my attitude towards such consignments. I would also like to thank the University of Allahabad and other universities included in my prototype for their consent to include their copyrighted pictures as a part of my project.

I thank all the support staff of department and the internet whose help directly and indirectly helped me complete my assignment.

Polamuri Mohana Krishna
B. Tech (6th Semester)
Computer Science & Engineering

# Abstract

Network Centric Warfare environment focuses on problem involving the location, characterization and identification of dynamic entities such as emitters, aircrafts, weapons, military unites. Multiple Target Tracking is a crucial requirement for surveillance system of interpreting the environment. In this work proposed a new view of Multiple Target Tracking with game theoretic perspective. The proposed new approach combines nearest neighbor method for tracking multiple targets with the Nash Equilibrium strategy of game theory. This method picks valid observation-to-track pair in optimized way and handle target origin uncertainties.


# 1. Introduction:

### 1.1 Purpose:

Multiple target tracking has application in both military and civilian areas. For instance, application areas include ballistic missile defense (reentry vehicles), air defense (enemy aircraft), air traffic control (civil air traffic), ocean surveillance (surface ships and submarines), and battlefield surveillance (ground vehicles and military units). The foremost difficulty in the application of multiple-target tracking involves the problem of associating measurements with the appropriate tracks, especially when there are missing reports (probability of detection less than unity), unknown targets (requiring track initiation), and false reports (from clutter). The key development of this project is to track multiple targets using one of the known algorithms for target tracking that is Nearest Neighbor approach combined with Game theoretic approach can be applied in that algorithm. We use Nash equilibrium of a mixed strategy game.

It is observed that the Nearest neighbour approach is simplest method in application of multiple target but not optimizing. Hence we used Hungarian algorithm to link the targets by analysing the cost produced by the munkres and generated a cost Matrix for every point of the target. We linked the observations of targets using optimised cost matrix generated by Hungarian algorithm with nearest neighbour method. We also applied nash equilibrium to see whether the provided cost matrix gives the best payoffs for the targets so that the error occurrence will be minimum. We got the values of the payoffs approximately equal to the cost of the targets and proved that we can apply game theory in Nearest Neighbour method.

### 1.2 Document Conventions

- ❖ The asterisks (*) are used denote that the specifications are particular.
- ❖ The digits from 1-9 are used to denote the priority among the data from high to low respectively.
- ❖ Items that are intended to stay in as part of your document are in **bold**
- ❖ Explanatory comments are in *italic* text.
- ❖ Plain text is used where you might insert wording about your project

### 1.3 Product Scope

The project Multiple Target tracking (MTT) we will basically deal with tracking the motion of the moving objects. Tracking, or particle linking, consist in re-building the trajectories of one or several objects as they move along time. Their position (2D co-ordinates (x, y)) is reported at each frame, but their identity is yet unknown: we do not know what object in one frame corresponding to an object in the previous frame. Tracking algorithms aim at providing a solution for this problem.

### 1.4 Technologies used:

- Language: MATLAB programming language.
- Platform Used: MATLAB 2013Rb
- MATLAB Script Files Technology

## 2. Definitions, Acronyms and Abbreviations:

**2.1 Data Association** is the procedure used to handle the measurement origin uncertainties. We can distinguish the following categories of data association problem in tracking:

- Measurement to measurement- track information (initiation)
- Measurement to tack- track maintenance (update or continuation).
- Track to track (in multi sensor situations) - track fusion.

Data association methods

- Nearest Neighbor (NN)
- Global nearest neighbor (GNN)

- Joint Probabilistic Data Association (JPDA)
- Multiple Hypothesis tracking (MHT)

**2.2 Nearest Neighbor** method is to find a predefined number of training samples closest in distance to the new point, and predict the label from these. The number of samples can be a user-defined constant (k-nearest neighbor learning), or vary based on the local density of points (radius-based neighbor learning). The distance can, in general, be any metric measure: standard Euclidean distance is the most common choice. Despite its simplicity, nearest neighbors has been successful in a large number of classification and regression problems, including handwritten digits or satellite image scenes. Being a non-parametric method, it is often successful in classification situations where the decision boundary is very irregular.
This is the cheapest (in term of accuracy) algorithm for linking that can be made. In particular, it is not guaranteed (and it is generally not the case) that the returned linking is an optimum for the sum of distances. Each source point is matched regardless of the others, there is no global optimization here (the Hungarian algorithm does that). Also, there exists refinement to nearest neighbor searches, such as the use of KD-trees; this contribution is exempt of such developments.

Below is the application of nearest neighbor algorithm.
These are the steps of the algorithm:

1. start on an arbitrary vertex as current vertex.
2. find out the shortest edge connecting current vertex and an unvisited vertex V.
3. set current vertex to V.
4. mark V as visited.
5. if all the vertices in domain are visited, then terminate.
6. Go to step 2.

The sequence of the visited vertices is the output of the algorithm.

**2.3 Hungarian algorithm** allows a "minimum matching" to be found. This can be used in instances where there are multiple quotes for a group of activities and each activity must be done by a different person, to find the minimum cost to complete all of the activities.

Hungarian Method - Pseudo-code

1. Subtract the smallest entry in each row from all the entries of its row.
2. Subtract the smallest entry in each column from all the entries of its column.
3. Draw lines through appropriate rows and columns so that all the zero entries of the cost matrix are covered and the minimum number of such lines is used.
4. Test for Optimality:
(i) If the minimum number of covering lines is n, an optimal assignment of zeros is possible and we are finished.
(ii) If the minimum number of covering lines is less than n, an optimal assignment of zeros is not yet possible. In that case, proceed to 5.
5. Determine the smallest entry not covered by any line. Subtract this entry from each uncovered row, and then add it to each covered column. Return to 3.

**2.4 Game theory** is "the study of mathematical models of conflict and cooperation between intelligent rational decision-makers." Game theory is mainly used in economics, political science, and psychology, as well as logic, computer science, biology and poker. Originally, it addressed zero-sum games, in which one person's gains result in losses for the other participants. Today, game theory applies to a wide range of behavioral relations, and is now an umbrella term for the science of logical decision making in humans, animals, and computers. The games studied in game theory are well-defined mathematical objects. To be fully defined, a game must specify the following elements: the players of the game, the information and actions available to each player at each decision point, and the payoffs for each outcome. A game theorist typically uses these elements, along with a solution concept of their choosing, to deduce a set of equilibrium strategies for each player such that, when these strategies are employed, no player can profit by unilaterally deviating from their strategy. These equilibrium strategies determine an equilibrium to the game—a stable state in which either one outcome occurs or a set of outcomes occur with known probability.

***Prescriptive or normative analysis of a Game***

Some scholars, like Leonard Savage, see game theory not as a predictive tool for the behaviour of human beings, but as a suggestion for how people ought to behave. Since a strategy, corresponding to a Nash equilibrium of a game constitutes one's best response to the actions of the other players – provided they are in (the same)

Nash equilibrium – playing a strategy that is part of a Nash equilibrium seems appropriate. This normative use of game theory has also come under criticism.

**2.5 Nash Equilibrium:** In game theory, the Nash equilibrium is a solution concept of a non-cooperative game involving two or more players in which each player is assumed to know the equilibrium strategies of the other players, and no player has anything to gain by changing only his or her own strategy. If each player has chosen a strategy and no player can benefit by changing strategies while the other players keep theirs unchanged, then the current set of strategy choices and the corresponding payoffs constitutes a Nash equilibrium.

Game theorists use the Nash equilibrium concept to analyse the outcome of the strategic interaction of several decision makers. In other words, it provides a way of predicting what will happen if several people or several institutions are making decisions at the same time, and if the outcome depends on the decisions of the others. The simple insight underlying John Nash's idea is that one cannot predict the result of the choices of multiple decision makers if one analyses those decisions in isolation. Instead, one must ask what each player would do, taking into account the decision-making of the others.

Nash equilibrium has been used to analyse hostile situations like war and arms races (see prisoner's dilemma), and also how conflict may be mitigated by repeated interaction (see tit-for-tat). It has also been used to study to what extent people with different preferences can cooperate (see battle of the sexes), and whether they will take risks to achieve a cooperative outcome (see stag hunt). It has been used to study the adoption of technical standards,[citation needed] and also the occurrence of bank runs and currency crises (see coordination game). Other applications include traffic flow (see Wardrop's principle), how to organize auctions (see auction theory), the outcome of efforts exerted by multiple parties in the education process, regulatory legislation such as environmental regulations (see tragedy of the Commons), analysing strategies in marketing and even penalty kicks in football (see matching pennies).

A game can have a pure-strategy or a mixed-strategy Nash equilibrium.
Nash's Existence Theorem is Nash proves that if we allow mixed strategies, then every game with a finite number of players in which each player can choose from finitely many pure strategies has at least one Nash equilibrium.

Step by step guide on how to find a Nash Equilibrium or Equilibria

Step 1. Look at the payoff matrix and figure out whose payoff's are whose:

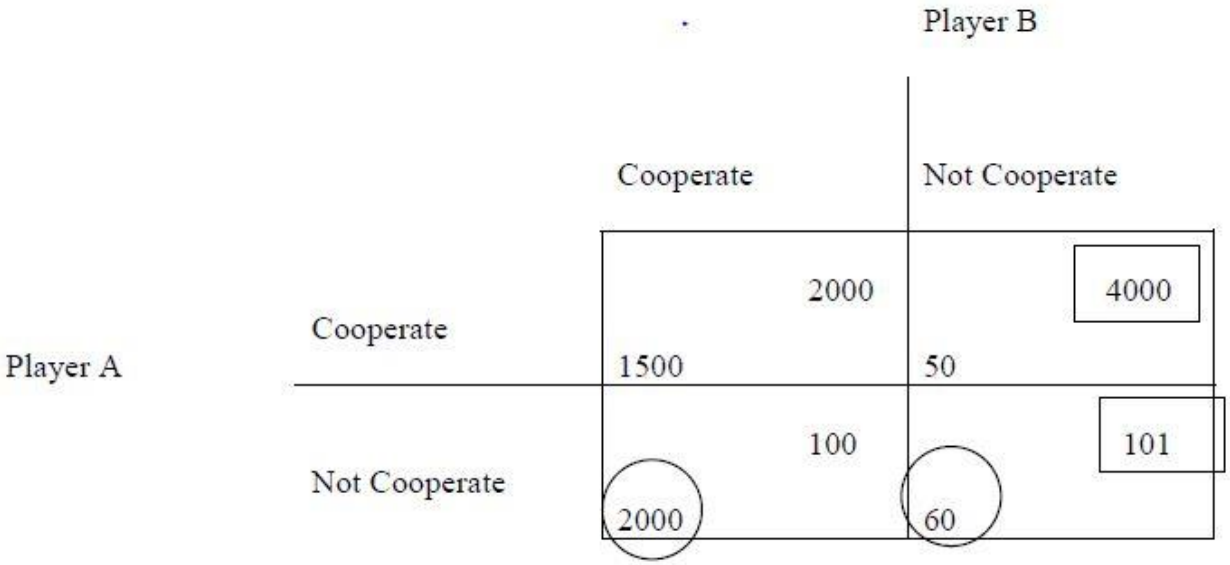
| | | Player B: Cooperate | Player B: Not Cooperate |
|---|---|---|---|
| Player A | Cooperate | 2000 (B)<br>1500 (A) | 4000 (B)<br>50 (A) |
| | Not Cooperate | 100 (B)<br>2000 (A) | 101 (B)<br>60 (A) |

Step 2. Figure out Player A's best response to all of player B's actions

| | | Player B: Cooperate | Player B: Not Cooperate |
|---|---|---|---|
| Player A | Cooperate | 2000<br>1500 | 4000<br>50 |
| | Not Cooperate | 100<br>2000 | 101<br>60 |

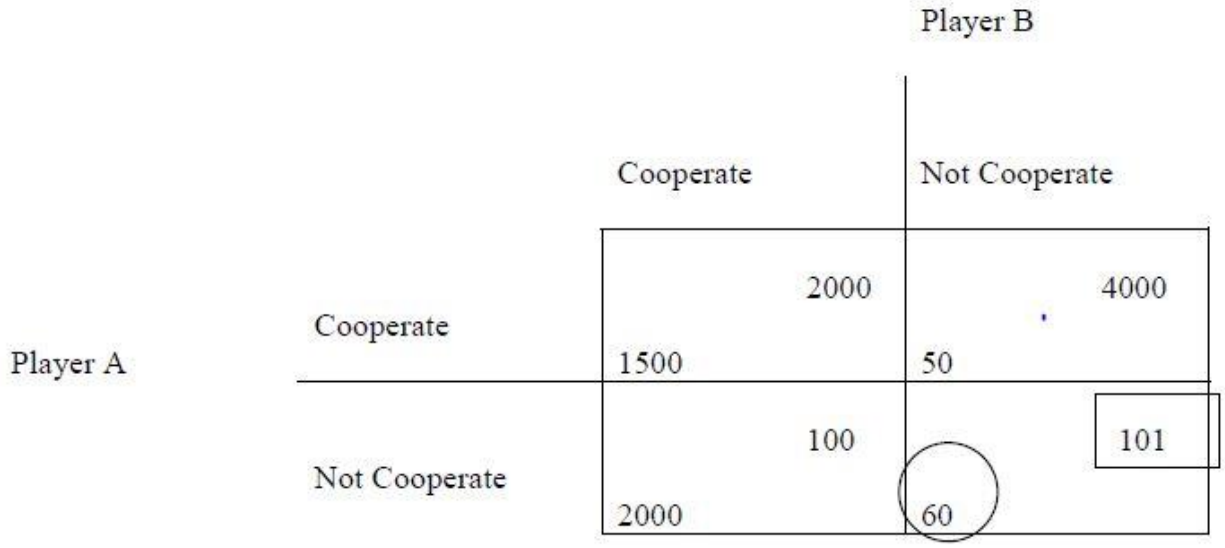
Since 2000 > 1500 and since 60 > 50

Step 3. Figure out Player B's best response to all of player A's actions




Since 4000 > 2000 and since 101 > 100

Step 4. A Nash equilibrium exists where Player B's best response is the same as Player A's best response



**2.6 MATLAB (Matrix Laboratory)** is a multi-paradigm numerical computing environment and fourth-generation programming language. A proprietary programming language developed by MathWorks, MATLAB allows matrix manipulations, plotting of functions and data, implementation of algorithms, creation of user interfaces, and interfacing with programs written in other languages including C, C++, Java, Fortran and other programming languages.

# 3. Goals and Planning

### 3.1 Time Plan

System Planning is an important for any successful project. Without proper planning the project is doomed. Good planning can be done after the requirements for the project are available. The input to the project planning activity is the requirement specification. The output of this phase is the project plan, which is the document describing the different aspects of the plan. The project plan is instrumental in driving the development process through the remaining phases. The major issues the project plan addresses are:

- Selection of technology.
- Development of modules.
- Cost Estimation//still to include
- Average Duration Estimation //still to include
- Gantt chart various models have been proposed for the software planning. E.g. COCOMO (**Co**nstructive **Co**st **Mo**del) developed by Boehm. The model fits the large scale projects and can be implemented with few modifications for the small projects.

### 3.2 Goals

The software which will be developed should have the following capabilities

- It should be able to track the targets.
- It should see the nearest possible neighbor for linking.
- If the target doesn't found any nearest point it must see the cost of the nearest point and link.
- It should record the points that target go through.
- After the generation of points it must give the cost matrix of the targets.
- It must be able to accept the payoffs of the cost matrix.

- The applied payoffs must give the correct payoff of the target by using Nash equilibrium.

# 4. Overall Description

### 4.1 Product Functions:

function [ target_indices target_distances unassigned_targets ] = nearestneighborlinker(source, target, max_distance)

target_indices = NEARESTNEIGHBORLINKER(source, target) finds for each point in 'source' the closest point in 'target'. These 2 inputs must be arrays with one point per row, and have their cartesian coordinates in each column (1D, 2D, 3D, ...). Nearest neighbor matching is based on euclidean distance. The two arrays might not have the same number of points.

The indices of the 'target' points are returned in an array 'target_indices', so that each row in 'source' matches the corresponding row in 'target(target_indices, :)'. The linking is exclusive: one source point is linked to at most one target point, and conversely. The linking is only locally optimal: the two closest points amongst the two sets are sought for first, then the second closest pair, excluding the first, etc... This ensures that the resulting linking will not depend on the order of the points in each set.

target_indices = NEARESTNEIGHBORLINKER(source, target, max_distance) adds a condition on distance. If the nearest neighbor is found to be at a distance larger than the given 'max_distance', they are not linked, and the 'target_indices' receive the value -1 for this source point. The same happens if all target points are exhausted.

[ target_indices target_distances ] =

NEARESTNEIGHBORLINKER(source, target)

additionaly return the distance to the matched target point. Un-matched

source points have a distance value set to NaN.

[target_indices target_distances unmatched_targets]=NEARESTNEIGHBORLINKER(source, target) additionaly return the indices of the points in 'target' that have not been linked.

function [ target_indices target_distances unassigned_targets total_cost ] = hungarianlinker(source, target, max_distance)

HUNGARIANLINKER link two lists of points based on the hungarian algorithm.

target_indices = HUNGARIANLINKER(source, target) finds for each point in 'source' the closest point in 'target'. These 2 inputs must be arrays with one point per row, and have their cartesian coordinates in each column (1D, 2D, 3D, ...). Source to target assignment is based on the famous hungarian algorithm using its excellent implementation by the excellent Yi Cao. The two arrays might not have the same number of points.

The indices of the 'target' points are returned in an array 'target_indices', so that each row in 'source'matches the corresponding row in 'target(target_indices, :)'. The linking is exclusive: one source point is linked to at most one target point, and conversely. The linking is globally optimal: the sum of the square distance is minimized, contrary to the naive nearest neighbor approach.

target_indices = HUNGARIANLINKER(source, target, max_distance) adds a condition on distance. If the nearest neighbor is found to be at a distance larger than the given 'max_distance', they are not linked, and the 'target_indices' receive the value -1 for this source point. The same happens if all target points are exhausted.

[ target_indices target_distances ] = HUNGARIANLINKER(source, target)

additionaly return the distance to the matched target point. Un-matched source points have a distance value set to NaN.

[ target_indices target_distances unmatched_targets ] =HUNGARIANLINKER(source, target)

additionally return the indices of the points in 'target' that have not been linked.

[ target_indices target_distances unmatched_targets total_cost ] = HUNGARIANLINKER(source,target)

additionally return the globally optimized value of the square distance sum.

function [assignment,cost] = munkres(costMat)

MUNKRES Munkres (Hungarian) Algorithm for Linear Assignment Problem.

[ASSIGN,COST] = munkres(COSTMAT) returns the optimal column indices,

ASSIGN assigned to each row and the minimum COST based on the assignment problem represented by the COSTMAT, where the (i,j)th element represents the cost to assign the jth job to the ith worker.

Partial assignment: This code can identify a partial assignment is a full assignment is not feasible. For a partial assignment, there are some zero elements in the returning assignment vector, which indicate un-assigned tasks. The cost returned only contains the cost of partially assigned tasks.

This is vectorized implementation of the algorithm. It is the fastest among all Matlab implementations of the algorithm.

function [A,payoff,iterations,err] = npg(M,U)

The function npg solves an n-person finite non-co-operative game to compute one sample Nash Equilibrium. It uses an optimization formulation of n-person non-co-operative games as described in the adjoining paper

"An Optimization Formulation to Compute Nash Equilibrium in finite

Games" presented by the author.

The inputs to the function are:

a) M : The row vector containing number of strategies of each player.

b) U : The matrix containing the pay-offs to the players at various

pure strategy combinations.

**4.2 Software Interfaces:**

- **Client on Internet:** Web Browser, Operating System (Windows 7, 8 and all other compatible operating systems)

- **Client on Intranet:** Client Software, Web Browser, Operating System (Windows 7, 8 and all other compatible operating systems)
- **Development End:** MATLAB 2013Rb
- **Tool Box Used:** Image Processing Toolbox available in MATLAB Version 8.2 with all its inherent (built-in) user-defined function files

### 4.3 Hardware Interfaces

- Processor: Intel® Core™i5-4200U CPU @2.3GHz
- System Type: 64-bit operating system, x64-based processor
- Ram: 6GB(5.89 GB usable)
- Hard disk: 1TB
- Keyboard: 105 standard

### 4.4 Constraints

***Design & Implementation Constraints:***

- GUI is only in English.
- This scripts generate random points that more or less follow a given trajectories. They are scrambled, their position is perturbed by noise, and some of them are randomly erased in some frames. This is the first and longer part in the script
- In a second time, the tracker tries to rebuild the tracks. This is made just by calling |simpletracker| with sensible parameters.
- The third part is only about displaying results. The points are drawn with a text label next to them indicating to which frame they belong. Track are drawn in color.
- MATLAB should be completely installed in the system along with the Image Processing Toolbox
- The MATLAB platform being used is only for a 64-bit processor henceforth a system with 32-bit processor or less will not be able to optimally implement and design the algorithms and codes devised.
- The histogram measurements of the subsequent watermarked image and its assessment and comparison to the original image using various parameters including PSNR, HOS can be performed only in the MATLAB interface.

***Memory Constraints***

Hardware memory: The programs are computationally very heavy and henceforth a sizeable amount of RAM (minimum 2 GB) for its efficient implementation in the MATLAB platform.

### 4.5 Adaptation requirements

No site adaptation is necessary in this project because the implementation of both target tracking and nash equilibrium is to be made portable. The entire system will have the capability to be transported to wherever it is needed. No external dependencies are in place and operation of the system will never change due to its varying locations.

# 5. System Requirements & Analysis

The following sections will introduce the numerous requirements of the system from the point of view of different users and will introduce a number of decisions that have been made regarding implementation. These sections also attempt to somewhat describe the role of each user group in the system, discussing their individual roles through the functions they can perform.

**User Interface**

No specific GUIs needed to be developed for this product. This is due to the fact that the program code developed for this project has been implemented in MATLAB which has a convenient built-in GUI (Figure Window, Command Window etc.) and has efficient interfacing functions which makes the use of GUI in this project futile.

**Performance Requirements:**

For the best performance requirement of the project we need efficient nearest neighbour linker and Hungarian algorithm for application and munkres for the reducing complexity of the target tracking. For application of nash equilibrium we need to give payoffs and strategies of the targets to know whether we can apply game theory in Target tracking.

**Learning Methodologies Adopted:**

- Familiarity with the domain: Worldwide research activities and the industrial interest in Multiple target tracking is increasing dramatically in today's world owing to the increasing popularity of internet. The field of object tracking is very vast. Understanding the interdependence and correlation of these fields was very crucial to my project. In this phase I studied about the real time

applications of Multiple target tracking and the ways in which people exploit the vulnerability of that on the net.

- Gather Information: With the help of various research papers and internet, we learnt the concepts of multiple target tracking techniques and different tracking algorithms. But these methods alone are not a solution to the problem because the noise in the real scenarios can make the solution go wrong so we need better application for this tracking.

- Learn MATLAB: MATLAB is a high level technical computing language and interactive environment for algorithm development, data visualization, data analysis, and numerical computation. Using MATLAB, you can solve technical computing problems faster than with traditional programming languages such as C, C++. One can use MATLAB in a wide range of applications, including signal and image processing, neural networks, communications, control design, test and measurement, and computational biology. Add-on toolboxes (collections of special-purpose MATLAB functions, available separately) extend the MATLAB environment to solve particular classes of problems in these application areas.

## 6. Software Quality Attributes

### Reliability

The system is to work reliably, with automatic point generation and tracking them. In case of unexpected error occurred in the tracking an acknowledgement will be given and will be took care immediately

### Availability

The entire system is being available round the year, except for a periodic maintenance. The maintenance period should be pre scheduled and short. The clients should be reminded of the unavailability period, well in advance.

### Maintainability

The documentation is to be made easy for the users who execute the systems day to day, for the developers who wish to edit or develop further, and for the personnel who is in charge of the maintenance.

### Portability

The program will be subsequently supported in newer, developed versions of MATLAB. The functions and methodologies used are standard and supported by most platforms.

**Usability**

The GUI is being easy to learn and use by users of any technical background. A built-in help feature in the MATLAB© programming platform is sufficient enough, to guide the users with the available functions and their functionalities. An easy to understand documentation is provided along with the system. System may support several languages because of inter-portable platforms available for translating MATLAB codes in to the required/desired language.

# 7. Experimental Results:

The proposed algorithm is tested on simulated 2-D co-ordinate data in MATLAB. The simulation is carried out with different scenarios consist of 4 targets moving with constant velocity and direction throughout the entire experiment. The result of tracking method in all scenarios shows that we get improved performance without any tracking errors and all target's tracks are assigned to correct observation.

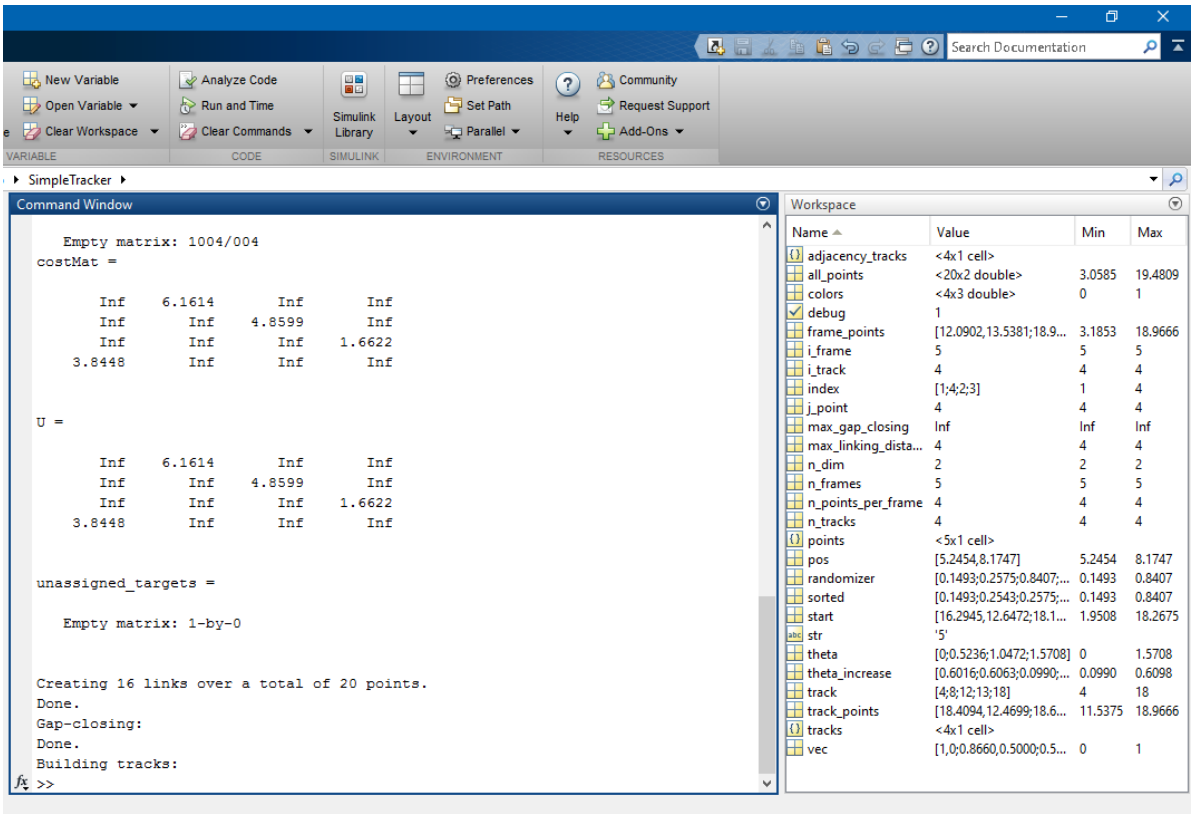



Figure 1. Output of tracking 4 targets

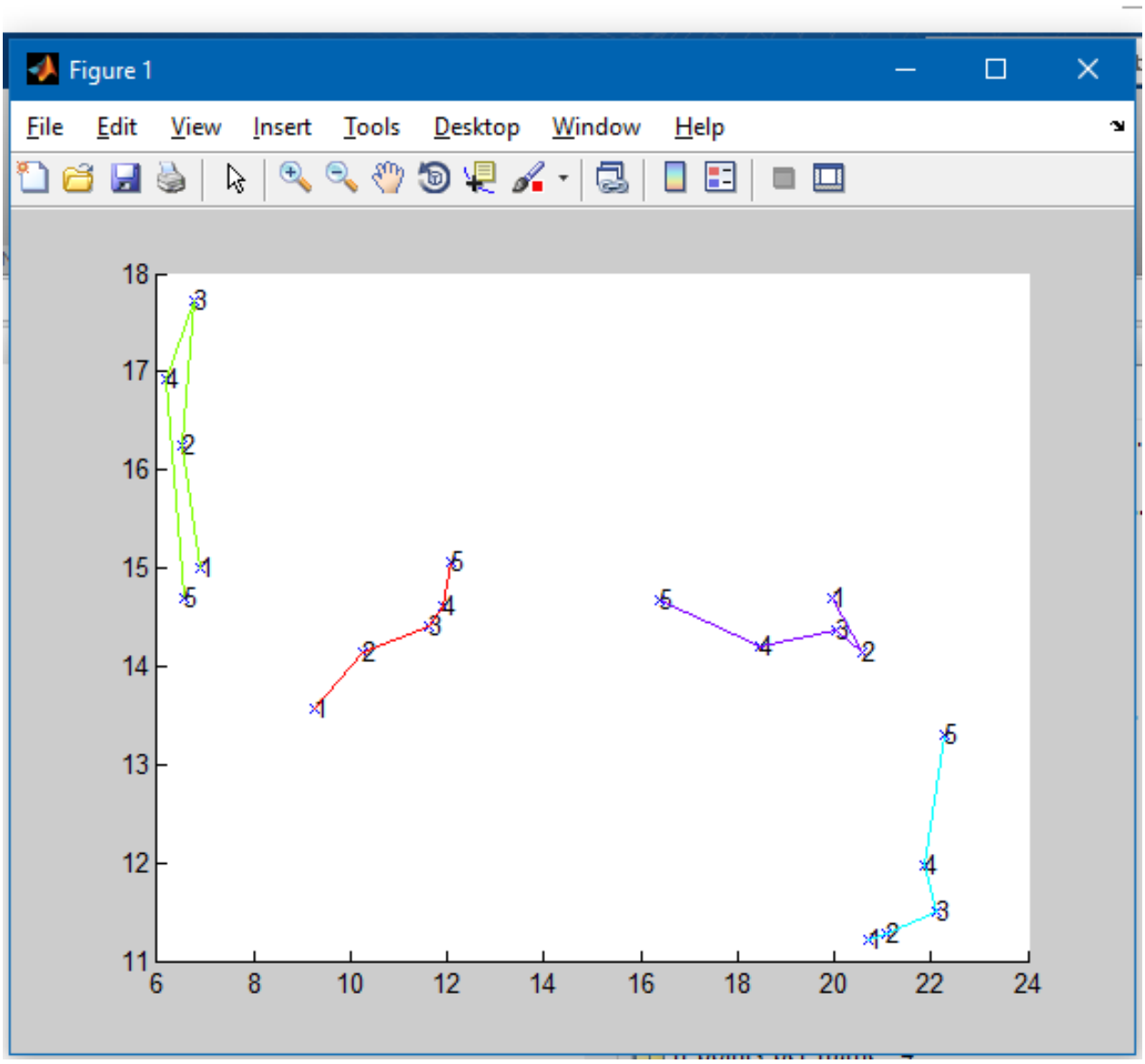



Figure 2. Tracking of 4 targets

4 targets have been tracked. No of unassigned targets will be 0 at the end of all iterations

## 2D of the targets

| 1 | | 2 | | 3 | | 4 | | 5 | |
|---|---|---|---|---|---|---|---|---|---|
| X | Y | X | Y | X | Y | X | Y | X | Y |
| 9.248594 | 13.57685 | 10.23464 | 14.13827 | 11.63225 | 14.40509 | 11.90285 | 14.61544 | 12.09329 | 15.06156 |
| 6.910965 | 15.01015 | 6.511121 | 16.24813 | 6.759348 | 17.72319 | 6.190584 | 16.91925 | 6.559963 | 14.69491 |
| 20.72063 | 11.22103 | 21.07639 | 11.28615 | 22.10025 | 11.51115 | 21.87114 | 11.97215 | 22.28118 | 13.29739 |
| 19.93944 | 14.69115 | 20.56953 | 14.14259 | 20.04782 | 14.36585 | 18.46053 | 14.20074 | 16.39023 | 14.67642 |

Target1: Red
Target2: Green
Target3: Blue
Target4: Violet

## For 3 Targets

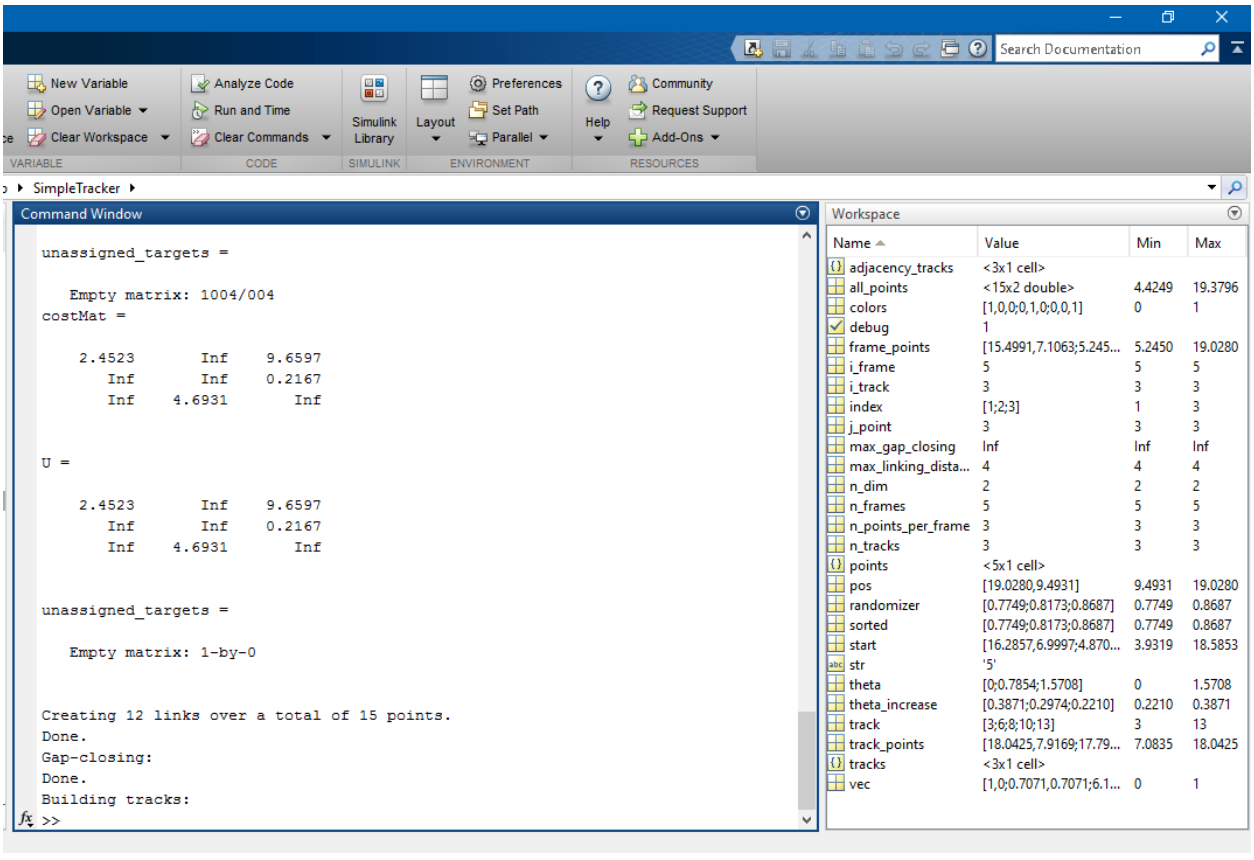



Figure 3. Output of tracking 3 targets

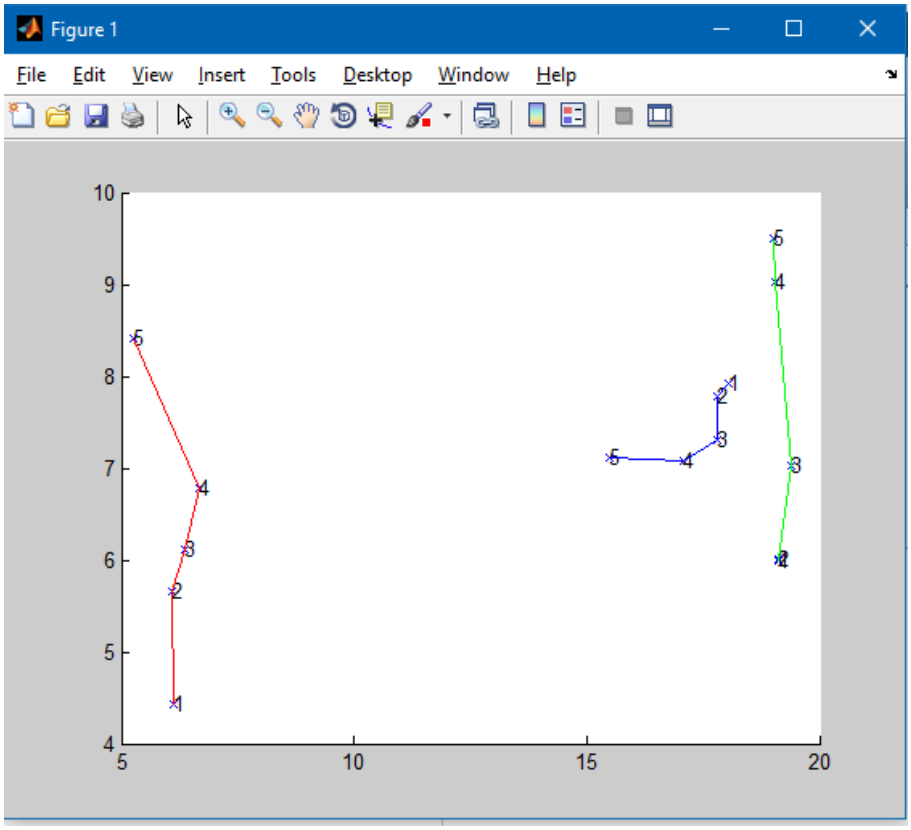



Figure 4. Tracking of 3 targets

## Application of game theory for 2 targets.

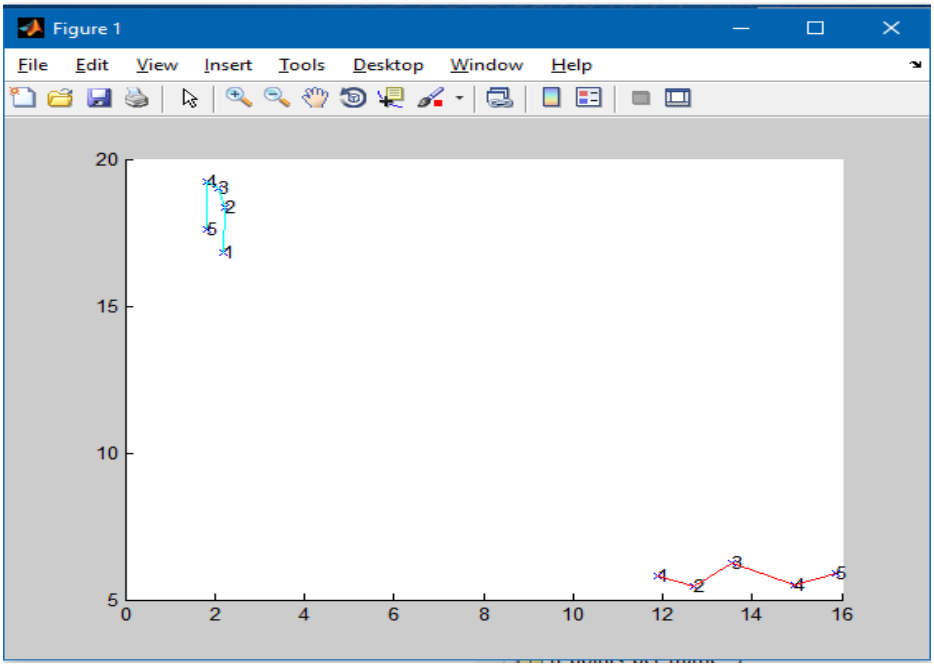


Figure.5 Tracking of 2 targets

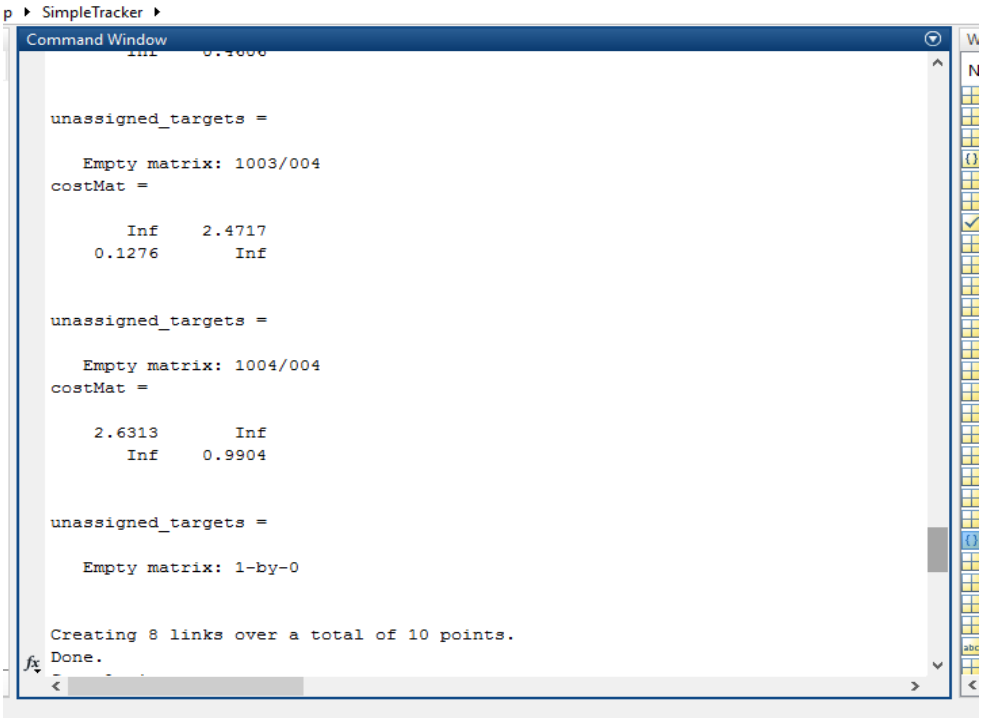
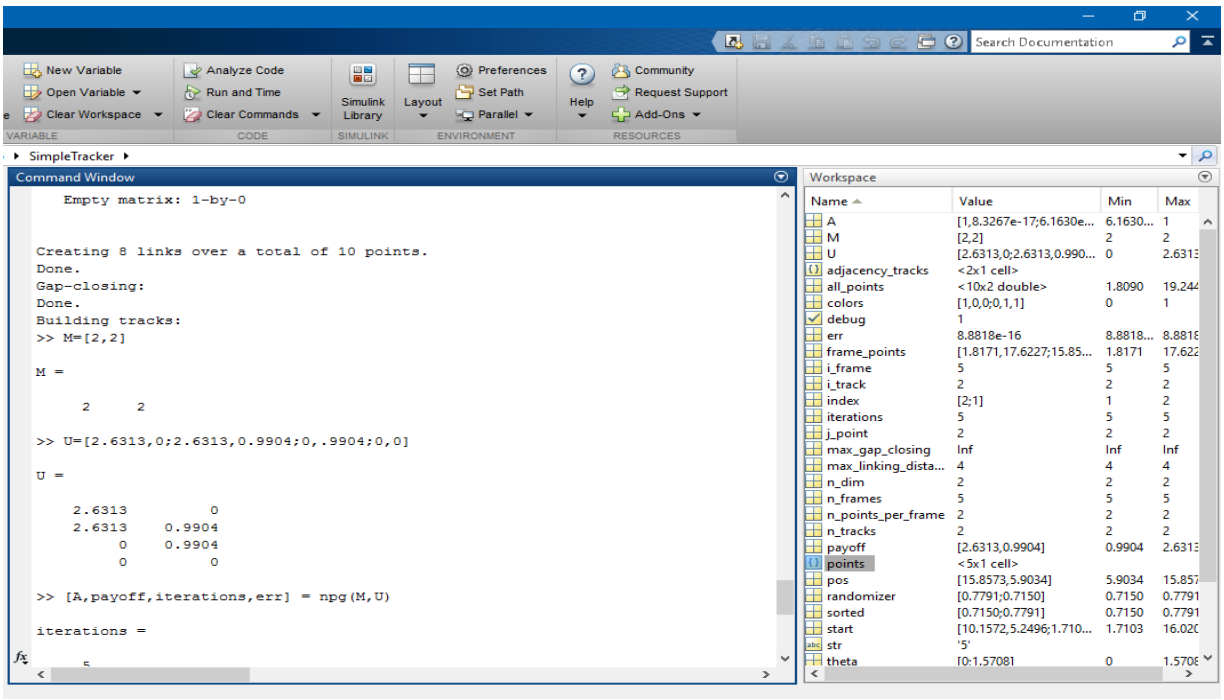



Figure 6. Result of tracking of 2 targets

Cost matrix of point 5 of both target are taken and read through nash equilibrium. They are 2.6313 of target 1 and 0.9904 for target 2



For the application of successful game the inputs of the game are given as shown in above picture. Inputs of the problem are explained at the product function in the document. Output of the game will be
A: The matrix whose columns are mixed strategies to players at Nash equilibrium.

payoff: The row vector containing the pay-offs to the players at Nash equilibrium.

iterations: Number of iterations performed by the optimization method.

err : The absolute error in the objective function of the minimization problem.

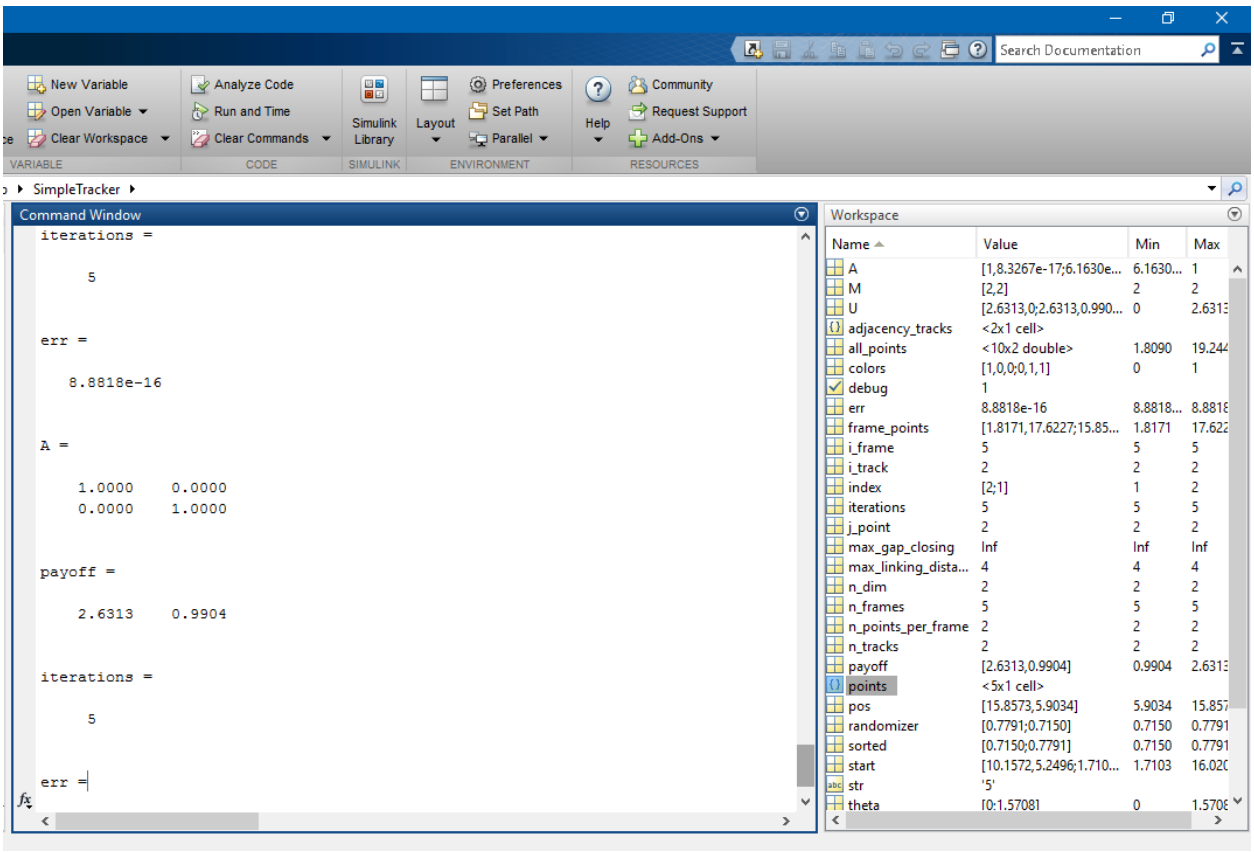



Figure 8.Payoffs for tracking 2 targets

# 9. Conclusion:

We have seen the performance of the tracker of multiple target and their behaviour in different scenarios. It is observed that the Nearest neighbour approach is simplest method in application of multiple target but not optimizing. Hence we used Hungarian algorithm to link the targets by analysing the cost produced by the

Munkres and generated a cost Matrix for every point of the target. Thus we linked that the observations of the target using optimized cost with nearest neighbour method. We then applied nash equilibrium to see whether the provided cost matrix gives the best payoffs for the targets so that the error occurrence will be minimum. We got the values of the payoffs approximately equal to the cost of the targets and proved that we can apply game theory in Nearest Neighbour. Game theoretic approach can also be applied in Probabilistic data association method.

# 10. References


1. Yakoov Bar-Shalom, Multi target- Multi Sensor Tracking: Advanced Applications, (Norwood, MA: Artech House,1990).
2. Bar-Shalom, Y., and X-R. Li, Multi target-Multi sensor Tracking: Principles and Techniques. Storrs, CT: YBS Publishing, 1995.
3. Blackman, S., R. Popoli, Design and Analysis of Modern Tracking Systems. Norwood, MA: Artech House,1999, ch.6, 7.
4. A. Yilmaz, O. Javed, and M. Shah, 2006. Object tracking: A survey. ACM Computing Surveys, vol. 38, no. 4, Article 13.
5. S. Blackman and R Popoli, 1999. Design and Analysis of Modern Tracking Systems. (Norwood MA, Artech House)
6. Thesis- "Multiple-Target Tracking in Complex Scenarios", Srinivas Phani Kumar Chavali, Washington University in St. Louis
7. M. Yang, T. Yu and others, "Game Theoretic Multiple Target Tracking", 2007 IEEE 11th international conference on compute vision, pp 1-8
8. Xiaolong Zhou, Y.F. Li and others, "Multi-Target Visual Tracking with Game Theory-Based Mutual Occlusion Handling", 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)