Summer 2011

Mutual Attraction Guided Search: a novel solution method to the Traveling Salesman Problem with vehicle dynamics

Jared Adam Nisbett

MUTUAL ATTRACTION GUIDED SEARCH: A NOVEL SOLUTION METHOD TO THE TRAVELING SALESMAN PROBLEM WITH VEHICLE DYNAMICS

by

JARED ADAM NISBETT

A THESIS

Presented to the Faculty of the Graduate School of

MISSOURI UNIVERSITY OF SCIENCE AND TECHNOLOGY

In Partial Fulfillment of the Requirements for the Degree

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

2011

Approved by

Dr. Donald C. Wunsch, Advisor
Dr. Randy H. Moss
Dr. Daniel S. Stutts

ABSTRACT

Traveling Salesman Problem (TSP) solution techniques are often used for route planning for automated vehicles. Most TSP solution methods focus on path length as the fitness reference, however in many cases, traversal time is of more practical importance. Mutual Attraction Guided Search (MAGS) is a novel solution method that uses an iterative process to simultaneously optimize both angle of travel through each target as well as the ordering of the targets in order to optimize path traversal time. MAGS deterministically locates a locally optimum solution quickly and can optimize for the acceleration limits of a specific vehicle rather than requiring a constant vehicle speed. Since the basic form of MAGS finds a solution deterministically, it has no mechanism for escaping local minima, therefore an evolutionary form is also developed that alternates between local search with MAGS and global search using evolutionary operators to combine and mutate solutions. This hybridization provides the necessary balance between local and global search that is required to locate a globally optimal solution. A fitness based on approximate travel time based on the maximum velocity achievable at each point on the path is calculated using the curvature of the path and the dynamic constraints of the vehicle. The performance of both the basic and evolutionary forms of MAGS are compared against path length based Euclidean and curvature constrained TSP methods.

ACKNOWLEDGMENTS

I would like to gratefully thank my advisor, Dr. Wunsch as well my committee members, Dr. Moss and Dr. Stutts for their advice, support, and encouragement. I would also like to thank Dr. Tauritz for devoting time to giving feedback and advice in the development of the evolutionary algorithm hybrid.

Support from the National Science Foundation, the Intelligent Systems Center, the Chancellor's Fellowship, and the M. K. Finley Endowment is gratefully acknowledged.

Finally, I would like to express my sincere thanks to my parents, for their guidance, wisdom, encouragement, and example throughout my life that have enabled me to reach this milestone.

TABLE OF CONTENTS

Page

LIST OF ILLUSTRATIONS

Figure Page

LIST OF TABLES

1. INTRODUCTION

Path and trajectory planning is a challenging task and is the subject of much research interest, especially as automation has been integrated into all parts of society including navigation and even vehicle piloting. Automated vehicles are increasingly being utilized by military and emergency response personnel to reduce risk to human lives. The practical cost of inefficient path planning and unnecessary exposure to hazards can be critical in situations where prolonged exposure may give the enemy time to target and destroy the vehicle. In some cases, the necessary trajectory is roughly defined by roadways and obstacles, especially for ground vehicles. In other cases though, particularly in cases of aerial vehicles, the trajectory is much more flexible and is defined largely by target waypoints. Path planning in such cases presents a challenge that is closely related to the Traveling Salesman Problem (TSP). The traditional Euclidean TSP has been widely researched as a path planning algorithm and is known in itself to be NP-hard [8]. Euclidean TSP solution methods can be used for route planning situations when targets are spaced far enough that the turning radius of the vehicle is negligible; however, if a cluster of targets are spaced on a similar scale as the vehicle's turning radius, the curvature constraints must be considered when planning the path.

For some vehicles, such as most airplanes and traditional four-wheeled vehicles, there is a minimum turning radius limit. The constraint may be due to the physical construction and placement of wheels on a car, or the maximum lift capabilities and stall speed of an airplane wing. For these types of vehicles, the curvature-constrained TSP must be solved. The curvature-constrained TSP is somewhat different from traditional TSP variants in that the actual path taken between nodes must be considered rather than simply using a table of distances.

Optimizing a curvature-constrained TSP path introduces several difficulties that are not present in many TSP variants as there are an infinite number of paths between any two points that can satisfy the basic curvature constraints of an arbitrary vehicle. A common method of simplifying the solution process is the use of the Dubins vehicle model - a vehicle that follows a path defined only by straight lines or circular arcs of a constant preset radius. This model has been proven to be optimal for a vehicle traveling at a constant speed. The derivation of a Dubins path is fairly straightforward since it is made up of circles tangent to the node angles and straight line segments tangent to those circles [1] as shown in Fig. 1.1. Only a few calculations are required to determine the shortest path possible with these constraints. However, even with this simplification, the problem is still more complex than traditional TSP since any change in the angle of travel through any node changes the optimal path to all other nodes and the problem is not scale invariant [4]. This is illustrated in Fig. 1.2 for the trivial case of a two-node problem in which the turning radius is limited to 1 unit and the nodes have a separation of two units. Figure 1.2(b) depicts the optimal solution. In Fig. 1.2(a), it is shown that even modifying only one node orientation can significantly affect the path length between nodes. The curvature-constrained TSP therefore cannot be reduced to a problem on a finite-dimensional graph as can many other TSP variants [10]. Thus, common combinatorial optimization tools cannot be

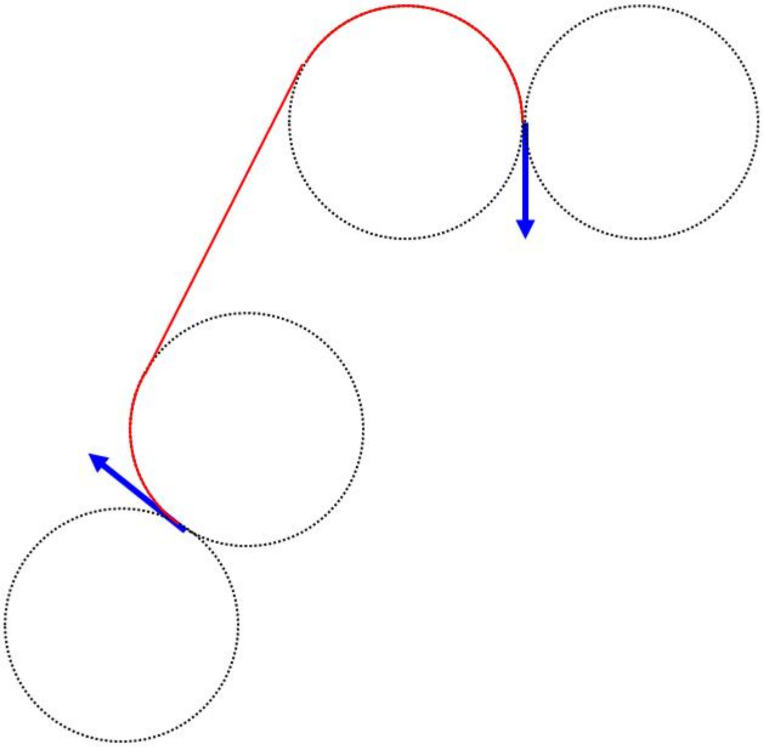


Figure 1.1. Example Dubins Path Construction

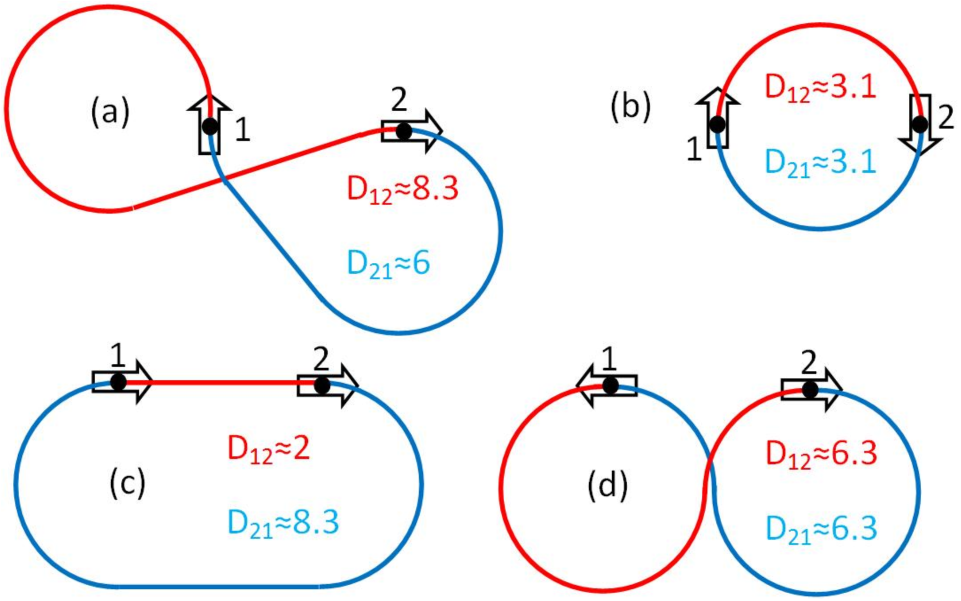


Figure 1.2. Path Length Dependence on Angle of Travel

easily applied. The TSP path planning problem becomes even more difficult when variable-speed vehicles are considered. The Dubins model can still be used to locate a path, however it is not guaranteed to be optimal and the minimum turning radius is not necessarily the most optimal radius, requiring an additional level of search in order to find the optimal radius.

Finally, certain vehicles do not have a minimum turning radius constraint at all. Helicopters are capable of making zero-radius turns if the speed of the vehicle is first brought to zero. These vehicles are capable of using Euclidean paths between points, however this is not necessarily optimal as it requires the vehicle to stop at every node. A simplistic example using only two nodes at unit separation is demonstrated in Fig. 1.3. In Fig. 1.4, the maximum velocity possible at each point along the paths is shown. For this example a tangential acceleration of 1 and a centripetal acceleration of 3 are used. Note that though the path length is increased for the paths with larger radius turns, the maximum velocity is also increased. The inverse of the velocities given in Fig. 1.4 are displayed in Fig. 1.5 with respect to the location along the path. The total time required to traverse the loop is represented by the area under each curve – found by taking the integral of the inverse velocity with respect

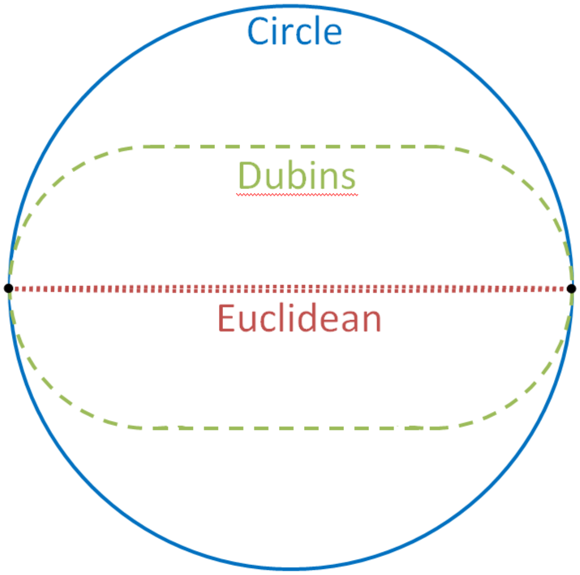


Figure 1.3. Example Paths for Traversal Time Comparison

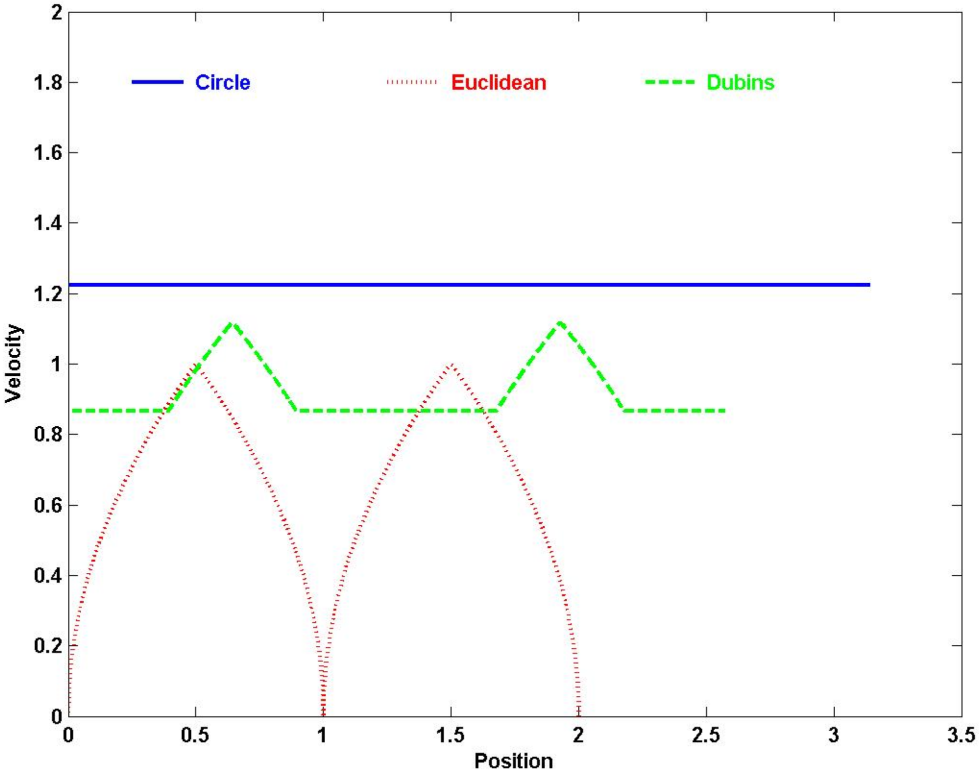


Figure 1.4. Velocity Profile Dependence on Turning Radius

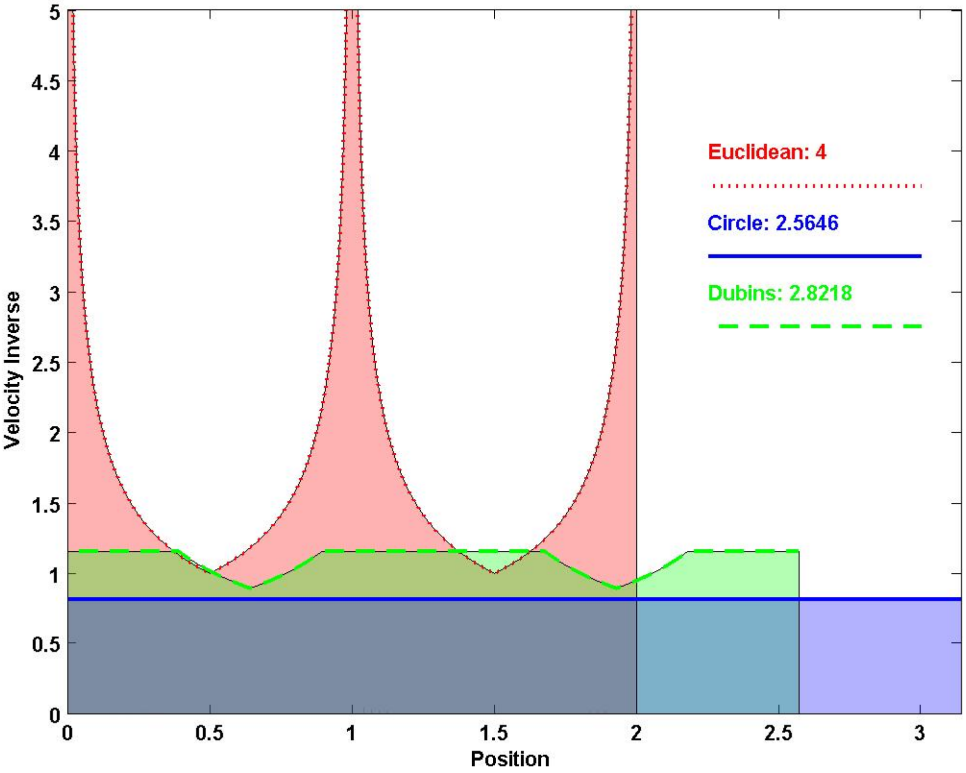


Figure 1.5. Path Traversal Time Dependence on Turning Radius

to time according to the relationship given in Eq. 1, which can be derived from the definition of velocity as the time derivative of position.

$$T = \int_{t_{begin}}^{t_{end}} dt = \int_{s_{begin}}^{s_{end}} \frac{1}{v}\, ds \tag{1}$$

In each graph, the red curve represents a vehicle following a straight line path between the nodes; The yellow describes a vehicle following a Dubins path with a radius of one-quarter of the distance between the nodes; Finally, the blue describes a vehicle following a circular path. The area under each curve is labeled. The relationship between the curves and the respective traversal time depends on the tangential and centripetal acceleration limits. In this case the circular path is optimal; however in other configurations the Euclidean or Dubins path may be optimal. The Euclidean path can be sufficient and even near-optimal for high tangential acceleration rates;

however, for lower acceleration rates, the Euclidean path is sub-optimal and so even if a vehicle is capable of making abrupt changes in direction it may not be desirable to limit it to Euclidean paths.

In essence, to obtain the optimal path through a set of nodes, a broader definition of the TSP must be used. Even in cases where a minimum curvature constraint is not applicable, the vehicle dynamics may play an important part in identifying an appropriate node ordering and trajectory shaping. In order to address many of these difficulties simultaneously, a new solution method, Mutual Attraction Guided Search (MAGS), is proposed which takes into account the dynamic constraints of a specific vehicle and identifies a continuous variable curvature path that attempts to minimize the vehicle's travel time. Since the MAGS solution method is deterministic, it works well for quickly finding a local minimum, however it becomes trapped by the nearest minima. Therefore, an evolutionary hybrid is further proposed in order to alleviate this problem and to facilitate global search. The Evolutionary - MAGS hybrid (E-MAGS) alternates between local optimization using MAGS and global exploration using combination and mutation operators.

2. RELATED WORK

The TSP has been a popular research subject and has been used in route planning for automated vehicles [4, 10, 6, 5, 7, 2, 11]. The TSP was only designed to address the ordering of the target nodes and does not consider the actual path necessary to move between points. Due to physical limitations, no vehicle is able to make radical changes in direction instantaneously. In many cases, particularly aerial vehicles, the dominating factor that restricts the radius of a turn is the forward momentum of the vehicle. A constant maximum normal force results in a radius bound that is proportional to the square of the velocity. In cases when the target waypoints are widely separated in comparison to the radius limit, the Euclidean TSP can still be used to plan the basic path, with adjustments to take into account the turning radius. However, when the waypoint density is comparable to the turning radius, the ordering of the targets as given by Euclidean TSP becomes sub-optimal [4]. Because of the exponential velocity relationship, increasing speed requires either handling much larger forces or using a larger turning radius. For a military surveillance vehicle, speed can be essential in order to avoid enemy fire; therefore it is not desirable to limit speed in order to make sharp turns.

Current applicable research is primarily focused on the Dubins TSP variant. Dubins first described the minimum path distance between points with prescribed initial and terminal tangent angles [1]. These curves are optimal when a vehicle's turning radius has a constant limit, either because the dominating factor is a physical constraint such as the steering mechanism, or because the dominant factor is the momentum and the vehicle maintains a constant speed. In order to deal with the additional complexity of the curvature constraints, current solutions either try to optimize node ordering and angle of travel separately – sometimes alternating between the optimization processes – or use an evolutionary process to make small changes to both node ordering and angle of travel. Often, only the angles are optimized and the ordering is assumed to be that of the Euclidean TSP solution [6, 7]. The author has not discovered any published research that uses a solution method which utilizes a non-stochastic search that optimizes both travel angles and node ordering simultaneously. The research using Dubins TSP variants is also sub-optimal in many cases for vehicles whose minimum turning radius is dependent on a variable speed. Tighter curves require a reduced speed, therefore Dubins solutions, while producing the shortest path for a given speed, either require choosing to use larger radii which increase path length, or require maintaining a slower speed in order to complete tighter turns after each node. In many practical applications, the traversal time is a more appropriate fitness metric than path length. The added complexity that must be faced by removing the Dubins curve assumption has discouraged exploration into minimum time solutions. In many cases however, a better solution would give preference to shallow curves when possible without significantly increasing the path length in order to allow the vehicle to traverse the path more quickly.

3. METHODOLOGY

3.1. THE MAGS ALGORITHM

The inspiration for MAGS lies in the interaction of bar magnets through attractive and repulsive fields. Each target node is modeled with a vector field similar to a bar magnet as shown in Fig. 3.1, though very simplistic in comparison to actual magnetic fields. The orientation of the node represents the angle at which the vehicle will pass through the node, with the vehicle passing from south to the north in the magnet illustration. A simplistic version of the solution process can then be conceptualized as if each node of the TSP was represented by a bar magnet centered at the node location and free to rotate about its center. If a set of magnets were thus arranged, they would each attempt to align themselves with the net field produced at their location by the other nodes. As the nodes rotate, the field adjusts accordingly until locally optimal magnetic loops are formed. MAGS utilizes a similar interaction in order to form path loops through the targets based on the vector field. The field strength produced by each node at a given location varies with respect to the relative position and the orientation of the node. Figure 3.2 depicts a node at position 1 at

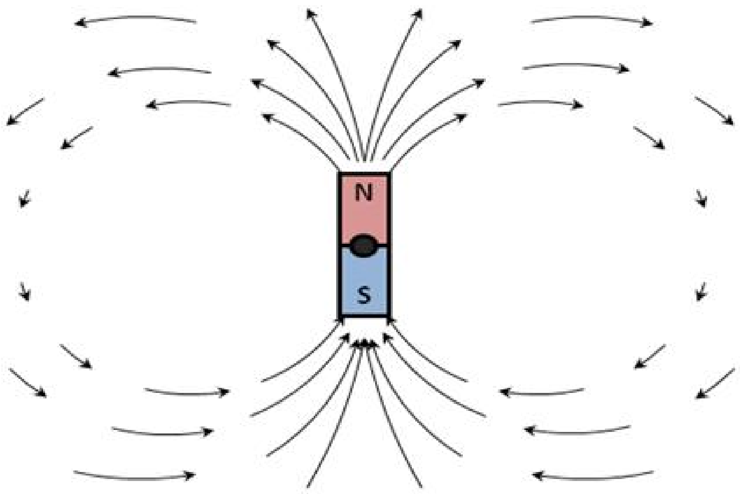


Figure 3.1. Conceptual Representation of Nodes in MAGS

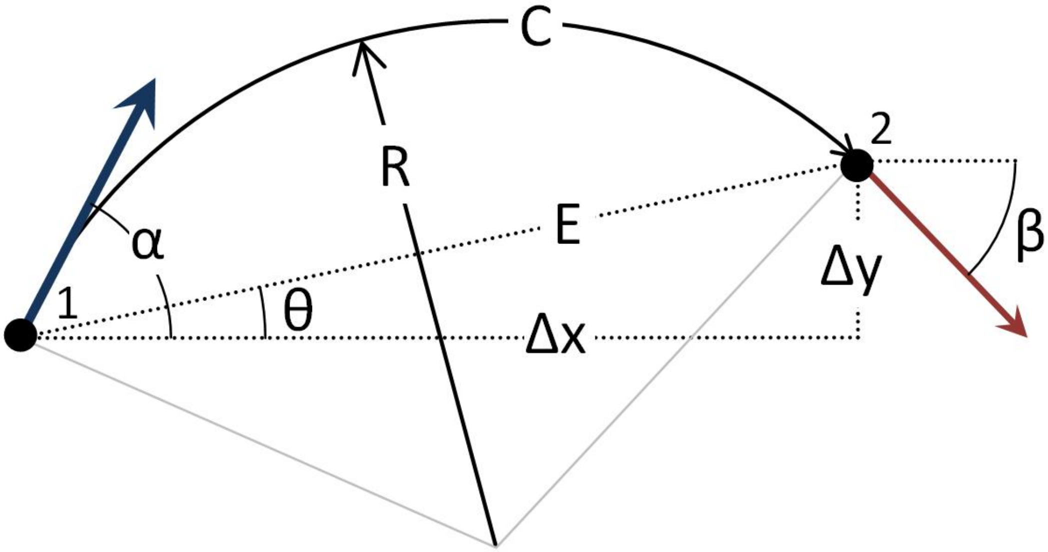


Figure 3.2. Derivation of Angle and Magnitude of Each Node's Contribution to the Vector Field at a Given Location

an angle of α. The field strength — represented by the red arrow at position 2 — is desired at some arbitrary position. The field strength at the desired location is calculated based on the properties of a circular arc which connects the node which is producing the field with the point of interest and which is tangent to the orientation of the node producing the field. In order to minimize travel time along the path, the proper balance between minimizing path length and maximizing the average radius of curvature must be achieved. Therefore the magnitude of the field produced at the point of interest is based on the arc length C and the radius R. These are labeled in Fig. 3.2 and calculated using Eqs. 2-3 where the euclidean separation E and relative orientation θ are given by Eqs. 4-5.

$$R = \left| \frac{E}{2 \cdot \sin(\alpha - \theta)} \right| \tag{2}$$

$$C = 2 \cdot R \cdot |\theta - \alpha| \tag{3}$$

$$E = \sqrt{\Delta x^2 + \Delta y^2} \tag{4}$$

$$\theta = \tan^{-1}\left(\frac{\Delta y}{\Delta x}\right) \tag{5}$$

The orientation of the field produced at the point of interest is tangent to the arc and is given by β in Fig. 3.2 and can be calculated using Eq. 6. In this study, the field strength was given by the relationship in Eq. 7 which is proportional to the traversal time of a circular arc at the maximum speed possible for that arc.

$$\beta = 2 \cdot \theta - \alpha \tag{6}$$

$$FieldStrength = \frac{R}{C^2} \tag{7}$$

Since MAGS essentially only identifies node ordering and angle of travel through each node, there is flexibility in how the actual path is derived from that information. Depending on the method chosen, the field strength relationship may be adjusted to give more influence to shallower curves or shorter path lengths. For each pair of nodes, there are four field strength relationships that are considered, as shown in Fig. 3.3. Two matrices were used to store the field strength data, one for "forward" data and one for "reverse" data. The "forward" fields are the fields generated by a given node that would be followed to travel to another node, while the "reverse" fields are the fields that would be used if traveling to the node generating the field from another node. The mutual connection between two nodes is then calculated by taking the average of the two fields that represent travel between the two nodes in a given direction. For example, to calculated the strength of the mutual connection from node A to node B, the average of the 'forward' field from A to B, and the 'reverse' field from B to A would be used. The distance and angle between nodes (given by E and θ in Fig. 3.2) are constant for a given problem and can be calculated once at the beginning of the algorithm. The basic solution process then consists of iteratively repeating the following steps until convergence:

1. Calculate the magnitude and direction of the net field produced at each node based on the current orientation of the other nodes.

2. Update the orientation of each node by adjusting it by some percentage of the difference between the current angle and the angle of the net field based on the magnitude of the net field.

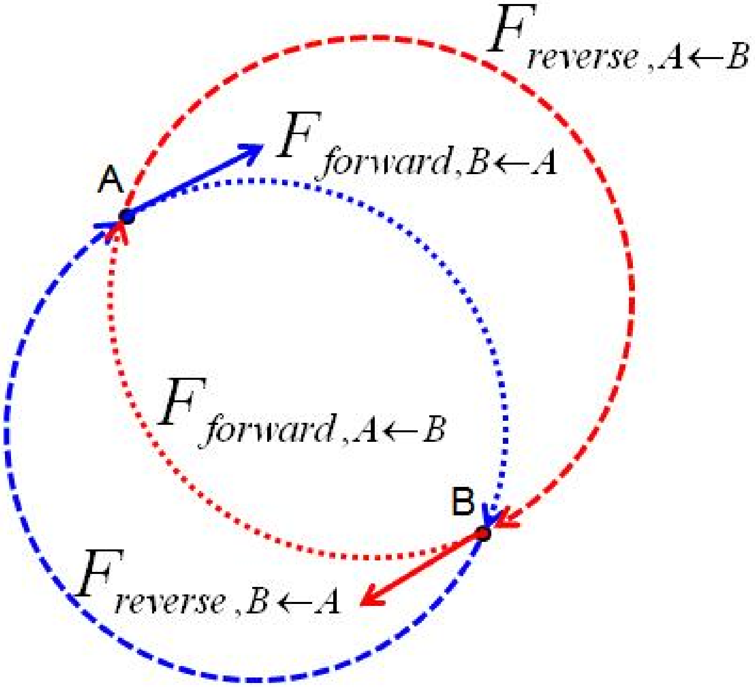
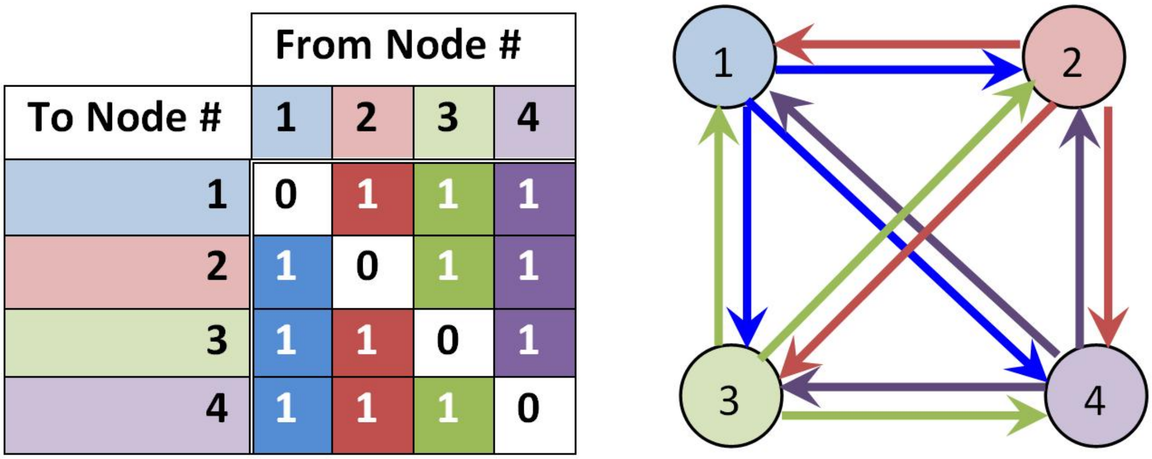

$F_{reverse,A \leftarrow B}$
$F_{forward,B \leftarrow A}$
A
$F_{forward,A \leftarrow B}$
B
$F_{reverse,B \leftarrow A}$

Figure 3.3. Field Connections Between Two Nodes

The amount that each node angle is adjusted to the net field angle is dependent on two factors. The movement of each node is scaled based on the magnitude of the net field normalized with respect to the greatest field strength present at any of the nodes at that instant. A damping factor of 0.7 was also applied so that no node moved more than 70% of the difference between the current state and the target state. This was added to help add stability and work towards a stable solution. Several considerations must be taken into account which add some further complexity to the procedure. In general, the magnitude of the field produced is directly proportional to the radius of the arc between two nodes. When the orientation of one node is directly in line with another node, however, an infinite radius results, therefore some maximum radius limit must be enforced before calculating the magnitude. A minimum radius limit may often also be enforced unless the vehicle is capable of a zero-radius turn. In the case of a minimum radius limit, the field magnitude is simply set to zero, while for the maximum radius limit, the radius is set to the maximum radius value and the arc distance is set to the Euclidian distance between the nodes. Additionally, the simple form of the algorithm only provides a solution in the form of node angles and a vector field between nodes. In order to quickly generate a path, it is necessary to also identify a node ordering. To accomplish this, a weight matrix representing the connectivity between nodes is used. At the beginning of the algorithm, all connection weights are set to 100% as illustrated in Fig. 3.4. After every iteration, a relative ranking of other nodes is generated for each node by scaling the magnitude of the individual field contributions based on the largest value. The connection between each pair of nodes is then updated based on the product of their rankings of each other. A connection adjustment rate can be used to tune the loyalty of the nodes. These connection weights are applied to the field magnitude contributions from each node in the next iteration when calculating the net field at each node. For each pair of nodes, there are also two connectivity weights to account for the direction of travel between the nodes (represented by the red and blue line segments in Fig. 1.2). Once the algorithm converges, the ordering of the nodes can be traced by identifying the highest ranking connection from each node - illustrated in Fig. 3.5.

Finally, in some cases, once the algorithm reaches convergence, multiple loops or isolated strings may have formed. For simplicity, these were combined by first identifying isolated strings and joining them into loops. Then, the mutual attraction of pairs of nodes in each loop (before application of the connection weights) is used to choose where to split and merge the loops. Once everything has been merged into



	From Node #			
To Node #	1	2	3	4
1	0	1	1	1
2	1	0	1	1
3	1	1	0	1
4	1	1	1	0

Figure 3.4. Connection Weight Matrix and Diagram with Full Connection

	From Node #			
To Node #	1	2	3	4
1	0	0	1	0
2	0	0	0	1
3	0	1	0	0
4	1	0	0	0

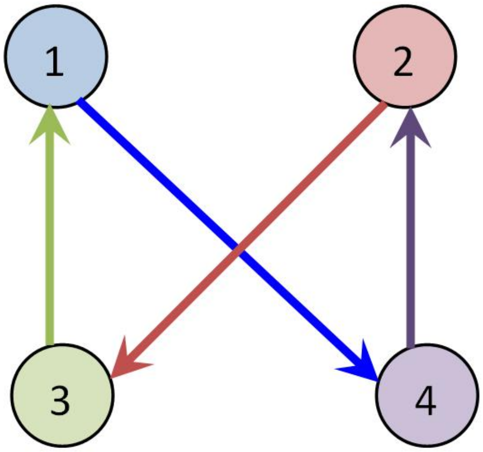


Figure 3.5. Connection Weight Matrix and Diagram for Single Loop

one large loop, MAGS is run for a few more iterations with the connection weights locked into the final node ordering in order to optimize the angles for that ordering. The full procedure within each cycle of MAGS is displayed as a flowchart in Fig. 3.6 and including the adjustment steps is then given as follows:

1. Calculate the raw magnitude and direction of the net field produced at each node based on the current orientation of the other nodes.

2. Adjust raw magnitudes.

   Correct for radii out of preset bounds

   Multiply by the connection weight matrix to yield adjusted fields.

3. Update the orientation of each node by adjusting it by some percentage of the difference between the current angle and the angle of the net field based on the magnitude of the net adjusted fields.

4. For each node, generate normalized rankings of all other nodes

5. Update connection weight matrix

6. Reduce loyalty

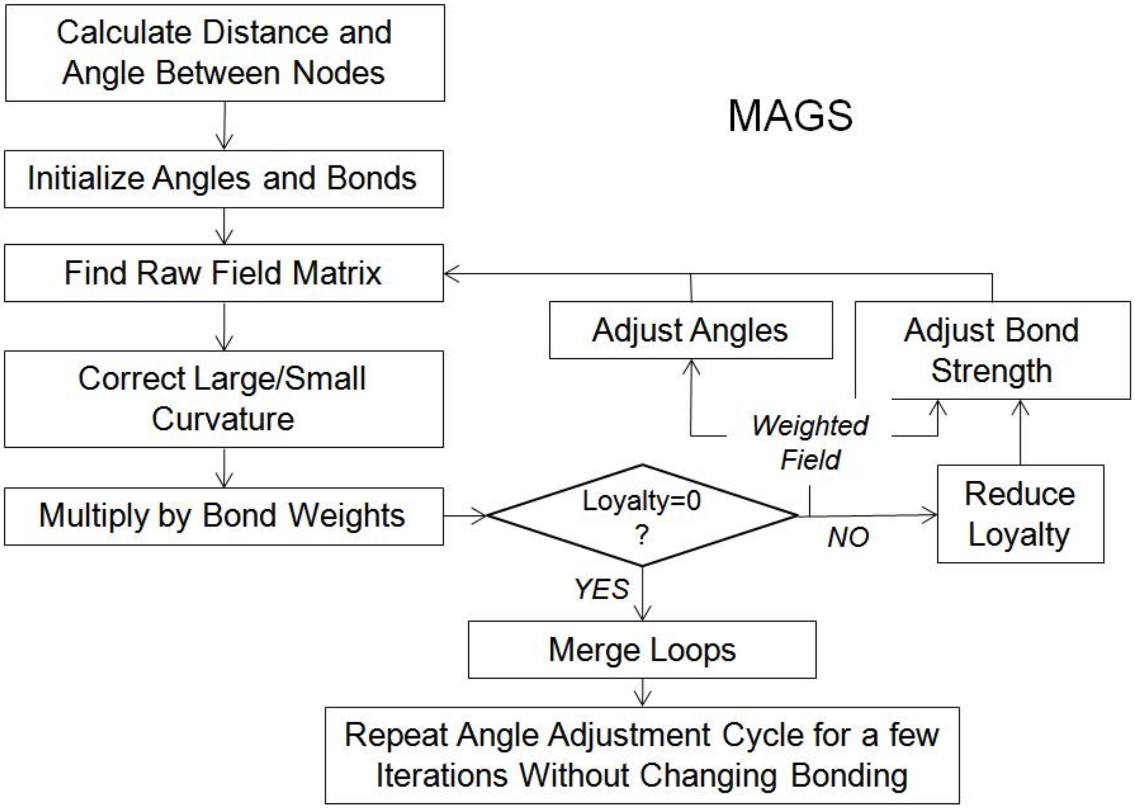

MAGS
Calculate Distance and Angle Between Nodes
Initialize Angles and Bonds
Find Raw Field Matrix
Correct Large/Small Curvature
Multiply by Bond Weights
Adjust Angles
Adjust Bond Strength
Weighted Field
Loyalty=0 ?
NO
YES
Reduce Loyalty
Merge Loops
Repeat Angle Adjustment Cycle for a few Iterations Without Changing Bonding

Figure 3.6. MAGS Process Flowchart

3.2. TRAJECTORY GENERATION AND FITNESS EVALUATION

The MAGS algorithm itself performs optimization based on the interaction of the vector fields generated by each node in order to find travel angles and ordering of the nodes based on the multi-objective goals of minimizing path distance while using shallower curves. The paths generated by MAGS will naturally tend towards optimal travel time, but the exact travel time for each curve is not needed in order to run the MAGS optimization process. However, in order to use evolutionary operators or to compare to other methods, an exact fitness of each solution candidate must be available, therefore an estimation of the actual travel time for each curve is necessary. There are many possible methods of generating a smooth curve between two points with initial and final tangents. Dubins curves have often been used and work well for constant speed vehicles, but the purpose of this model is to allow more flexibility in the path rather than requiring a constant turning radius. Some researchers have also used various spline curves with success. The search for the optimal curve is a complex problem and merits much study in itself. In general, more optimal trajectories require more computation time to locate and since this study requires many repeated fitness analyses, a simple model was necessary. It is likely that other curve generation methods may yield better results at the expense of time and further research may even yield more efficient curve generation methods.

The derivation of the paths used to determine fitness is illustrated in Fig. 3.7. Two circular arcs are used as the basis, each tangent to the angle of one node and passing through the other node. Parametric equations describing each arc are given in Eq. 8-9 using normalized scaling with parameter t such that a constant speed of motion along the arc from the first point to the second is given by varying t from 0 to 1 at a constant rate. Note that the parameter t represents a normalized path parameter that gives the percentage of completion of the path from one node to the next and does not represent time.

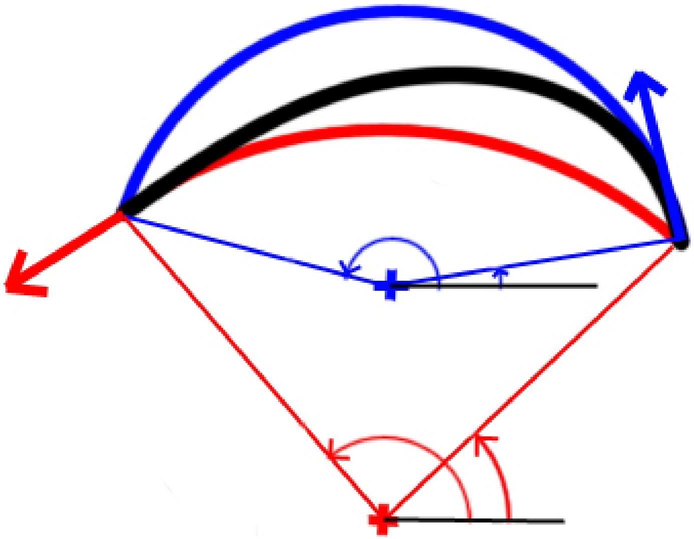


Figure 3.7. Derivation of Exact Path Between Nodes

$$x_i(t) = X_{c,i} + R_i \cdot cos((\theta_{i,first} - \theta_{i,second}) \cdot t + \theta_{i,first}) \tag{8}$$

$$y_i(t) = Y_{c,i} + R_i \cdot sin((\theta_{i,first} - \theta_{i,second}) \cdot t + \theta_{i,first}) \tag{9}$$

The point $(X_{c,i}, Y_{c,i})$ is the center of the circle, R_i is the radius of the circle, $\theta_{i,first}$ is the start point, and $\theta_{i,second}$ is the end point. The path used is based on gradually adjusting from one circle to the other based on the parameter t as given in Eq. 10.

$$x_{path}(t) = x_1(t) \cdot (1 - t) + x_2(t) \cdot t \tag{10}$$

The maximum velocity achievable on a given path segment is given by Eq. 11 where ρ is the radius of curvature and a_n is the maximum normal acceleration of which the vehicle is capable.

$$V_{max} = \sqrt{a_n \cdot \rho} \tag{11}$$

The radius of curvature, ρ at any point on a parametric curve is based on the first and second derivatives of the parametric equations as given in Eq. 12.

$$\rho = \frac{(x'^2 + y'^2)^{\frac{3}{2}}}{x'y'' - y'x''} \tag{12}$$

From Eq. 8-10 the first and second derivatives of x_{path} and y_{path} can be calculated as given in Eq. 13-16 where C_i, S_i and $\Delta\theta_i$ are defined as given in Eq. 17-19.

$$x'_{path} = -S_1 \cdot \Delta\theta_1 \cdot (1 - t) - S_2 \cdot \Delta\theta_2 \cdot t + X_{c,2} - X_{c,1} + C_2 - C_1 \tag{13}$$

$$x''_{path} = -S_1 \cdot \Delta\theta_1^2 \cdot (1 - t) - 2 \cdot C_2 \cdot \Delta\theta_1 - S_2 \cdot \Delta\theta_2^2 \cdot t + 2 \cdot C_2 \cdot \Delta\theta_2 \tag{14}$$

$$y'_{path} = C_1 \cdot \Delta\theta_1 \cdot (1 - t) + C_2 \cdot \Delta\theta_2 \cdot t + Y_{c,2} - Y_{c,1} + S_2 - S_1 \tag{15}$$

$$y''_{path} = -C_1 \cdot \Delta\theta_1^2 \cdot (1 - t) + 2 \cdot S_1 \cdot \Delta\theta_1 - C_2 \cdot \Delta\theta_2^2 \cdot t - 2 \cdot S_2 \cdot \Delta\theta_2 \tag{16}$$

$$\Delta\theta_i = \theta_{i,first} - \theta_{i,second} \tag{17}$$

$$C_i = R_i \cdot cos(\Delta\theta_i \cdot t + \theta_{i,first}) \tag{18}$$

$$S_i = R_i \cdot sin(\Delta\theta_i \cdot t + \theta_{i,first}) \tag{19}$$

The maximum velocity at each parametric curve point generated can then be identified. In order to calculate the travel time, it is first necessary to calculate the actual velocity achievable at each point, taking into consideration the time needed to accelerate and decelerate in order to meet velocity requirements at other points. The tangential acceleration limit of the vehicle is the limiting factor. Starting with the absolute maximum velocity profile for the curve, two adjustment passes are run using Eq. 20 to determine the change in velocity possible at the given acceleration limit. The initial velocity is given by v_i, a_t represents the tangential acceleration limit of which the vehicle is capable, and Δs is the distance moved along the curve.

$$v = \sqrt{v_i^2 + 2 \cdot a_t \cdot \Delta s} \tag{20}$$

The first adjustment pass works forward along the curve and adjusts any velocities that cannot be achieved with the set acceleration limit and the velocity at the last point. The second adjustment pass works in reverse and ensures that the velocity at each point is not too high to decelerate to the required velocity by the next point. Once the actual velocity is found for each point on the curve, the total travel time can be found by integrating the inverse of the velocity with respect to curve position as given in Eq. 21.

$$T = \int_{t_{begin}}^{t_{end}} dt = \int_{s_{begin}}^{s_{end}} \frac{1}{v}\, ds \tag{21}$$

The quality of the travel time approximation can be controlled by changing the number of points used in the discrete parametric approximation. It should be noted that the nature of the parametric equation will not yield equally spaced points. The generated points will tend to be more densely spaced near each end of the curve in most cases. The difference in density depends on how well the orientation angles of the two nodes are in harmony. When there is a large disparity between the preferred curves, the density of the generated points will tend to be more widely varied while generated points will be more evenly spaced when the orientation angles of each node are in alignment with the preferred curve from the other node.

3.3. E-MAGS HYBRID

The basic structure of the evolutionary hybrid is shown in Fig. 3.8. In developing the EA hybridization, the primary focus was in the development and analysis of appropriate recombination and mutation methods. Any standard parent selection operators can be used, therefore the analysis of optimal selection operators was not investigated in this research. A simple *k*-tournament selection process with replacement was therefore used in the selection of two parents for each new offspring in all tests. A *k*-tournament selection is performed by randomly selecting a set number, *k*,

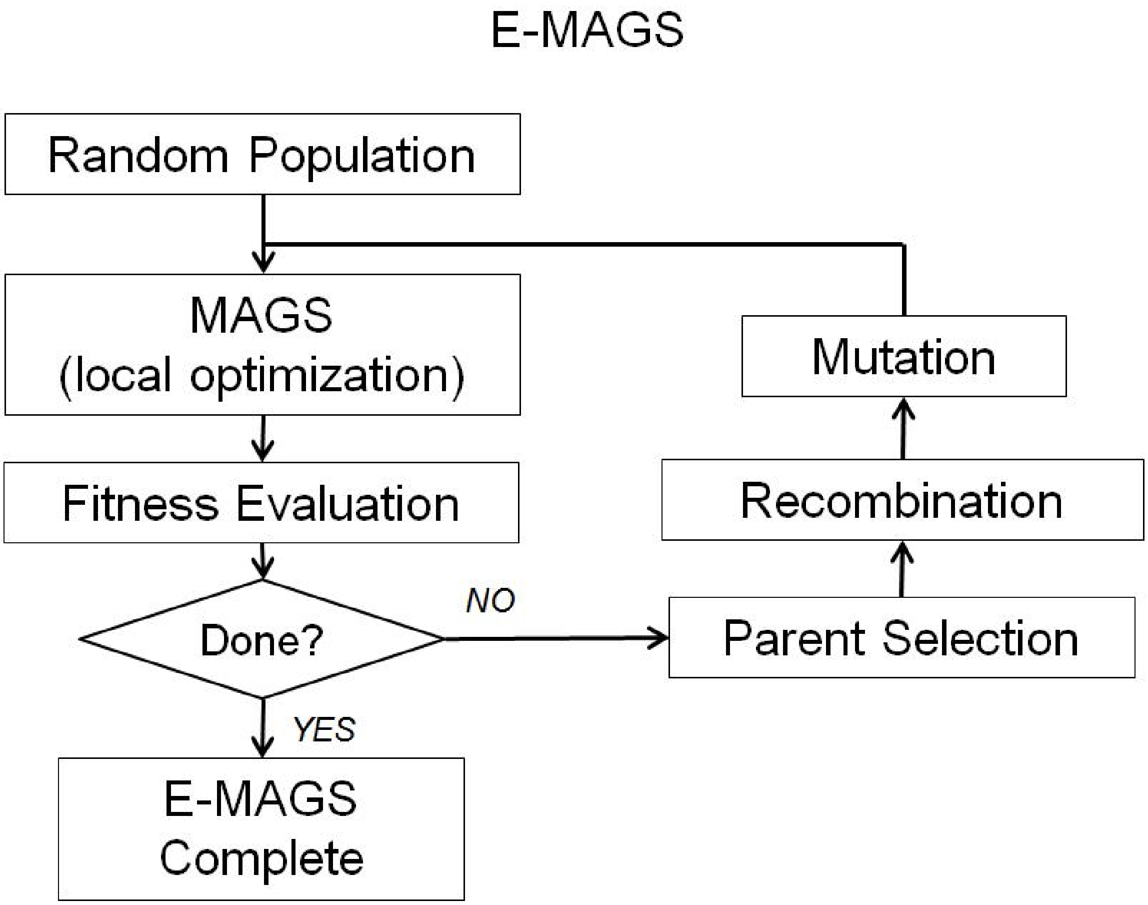


Figure 3.8. E-MAGS flowchart

of individuals for each tournament. The fittest individual within the selected group is chosen to be a parent of a new offspring. Two tournaments are held for each offspring in order to generate both parents. Once a new population was created, the parent generation was discarded. The recombination and mutation methods available depend on the encoding format used to represent the individual solutions. All curvature constrained TSP solutions are made up of two elements: travel angles, and node ordering. The travel angles were encoded as a vector of length n,where n is the number of nodes, and the node ordering was encoded in the form of the connection weight matrix as illustrated in Fig. 3.5. For the recombination stage, standard operators may be used for the angle vectors. A random crossover was utilized with each angle being selected from either parent with equal probability. Several variations were tested for the recombination of the weight matrices. The simplest option tested was to simply ignore the node ordering information and completely reset all connection weights so that each new individual starts with full connectivity between nodes as shown in Fig. 3.4. The second option tested was an element-wise OR operation that combines the connections from both parents. This is illustrated in Fig. 3.9 for parents with loops of 1-4-2-3 (Fig. 3.5) and 1-2-3-4. Finally, an intermediate method was tested - starting with the element-wise OR operation, and then partially resetting the remaining weights as shown in Fig. 3.10.

	From Node #			
To Node #	1	2	3	4
1	0	0	1	1
2	1	0	0	1
3	0	1	0	0
4	1	0	1	0

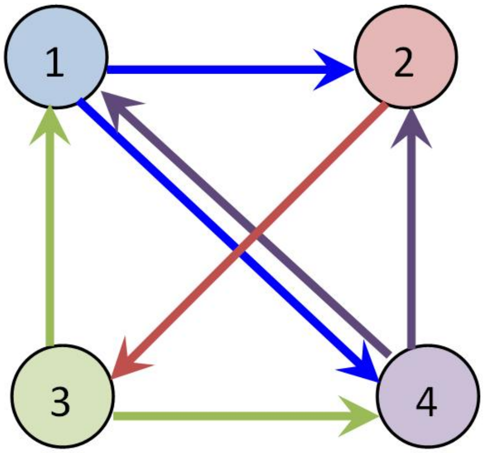
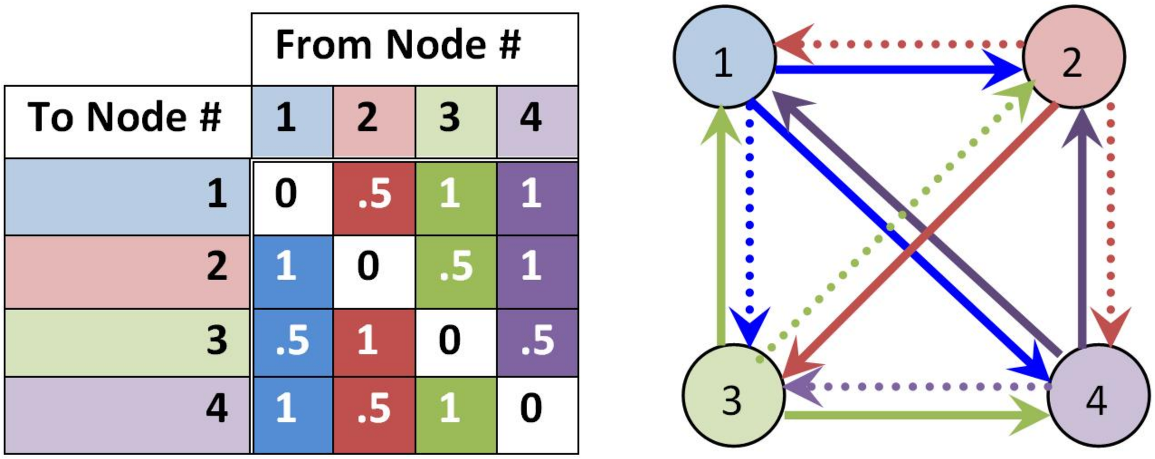


Figure 3.9. Connection Weight Matrix and Diagram for Element-wise OR Recombination with no Reset



	From Node #			
To Node #	1	2	3	4
1	0	.5	1	1
2	1	0	.5	1
3	.5	1	0	.5
4	1	.5	1	0

Figure 3.10. Connection Weight Matrix and Diagram for Element-wise OR Recombination with Partial Reset

Mutation of the angle vectors was performed by resetting each angle to a random value with a set probability. If and only if the angle of a node was reset, all weight connections to and from that node were reset as well.

3.4. BASELINE COMPARISONS

In order to obtain a meaningful performance analysis, a baseline comparison is needed. The simplest comparison used was that of the Euclidean path. Using the maximum velocity and tangential acceleration limits given, the traversal time was calculated for the optimal Euclidean path. The vehicle was assumed to be able to change direction abruptly when at zero velocity. While this path may not be feasible for all vehicles, it provides a meaningful comparison. Secondly, a Dubins path comparison was calculated. For simplicity, the node ordering given by the Euclidean TSP was used and angles were optimized using Particle Swarm Optimization [3]. The experiment was repeated using a range of radii for the Dubins paths and the best radius was used for comparison against the results of MAGS and E-MAGS. Even though Dubins paths were used, a constant velocity was not enforced, but rather the vehicles were assumed to have the same acceleration and velocity limits as for the other methods and the velocity profile was generated accordingly.

4. EXPERIMENTAL DESIGN

Experimental testing was carried out using MATLAB on a standard desktop computer. All experiments were conducted with ten independent runs for each configuration tested in order to measure statistical significance of the results. The number of independent runs was limited to ten because of the number of experiments necessary and the computation time required for each experiment. The Burma14 and Ulysses16 benchmark datasets from the TSPLIB collection [9] were used to test performance. While these are fairly small TSP datasets, the computational complexity required the use of smaller datasets for initial testing. Additionally, the advantages of using MAGS are most effective on tight clusters of nodes. For larger datasets, much of the dataset may be able to use Euclidean ordering to simplify the problem. Ideally, dense clusters could be identified and solved with MAGS and then integrated into the overall solution.

In order to allow a meaningful comparison between the two datasets, all velocity and acceleration limits were scaled based on the average separation between nodes. Centripetal acceleration limits were tested at 1, 3, and 5 times the average separation. For the Burma14 dataset the tangential acceleration was tested at both 1 and 5 times the average node separation and the maximum velocity was set to 3 times the average separation. For the Ulysses16 dataset, a tangential acceleration multiplier of 1 was used and the maximum velocity was tested at both 3 and 5 times the average separation. As a scale of reference, for a minimum node separation of 100 ft, a velocity multiplier of 3 and 5 would correspond to a maximum velocity of about 200 mph and 340 mph respectively. Alternatively, a minimum node separation of 1000 ft would correspond to maximum velocities of around Mach 3 and Mach 5. Acceleration multipliers of 1, 3, and 5 correspond to 3g, 9g and 15g for an average node spacing of 100 ft - ranging from human discomfort to loss of consciousness limits.

Preliminary testing of MAGS showed only marginal improvement with more than 80-100 cycles, so MAGS was run for 100 cycles for each individual in the population while generating initial solution candidates from a random start configuration. When using E-MAGS, new offspring generated through evolution were run for only

50 cycles within MAGS since the solution should be partially optimized at the beginning of the MAGS cycle. E-MAGS was run for 30 generations. When testing the basic MAGS algorithm, every new run is started randomly, so generations and population sizes are unnecessary. In order to gain an effective comparison to the E-MAGS performance, 30 generations of equal population were also used, however every individual in each generation was simply initialized with new random starting angles. However, all generations were run for the full 100 cycles. A population size of 15 and a tournament size of 4 were selected. The computational complexity of MAGS placed limits on the practicality of testing larger population sizes.

In each configuration, E-MAGS was tested with mutation rates of 0.0, 0.2 and 0.4 and connection weight reset percentages of 0%, 50%, and 100%. In order to compare with the performance of MAGS, a comparison run of MAGS was performed for each acceleration value in which an equal population was used, but all members of the population were reset with new angles and full connectivity with each generation.

For the Dubins path tests, Particle Selection Optimization (PSO) [3] was used to optimize the angles of travel through each node. Twenty different radii were tested in each configuration and the best selected for comparison with the other methods. The radii used were based on the radius achievable while traveling at the maximum velocity. Early testing indicated that the optimum radius was typically found to be less than 1/100th of the maximum radius possible so radii were tested between 0.000025 and 0.01 times the maximum radius. The PSO parameter constants were each set to 2 and a maximum particle velocity was set to $\frac{\pi}{2}$. One hundred iterations were run for each configuration.

5. RESULTS

Table 5.1 displays the best fitness after 25 generations for each parameter configuration. The values displayed represent the average value over 10 runs. The variances associated with each of these runs are displayed in Table 5.2. Tables 5.3 and 5.4 list the mean and variance of the fitness on the Ulysses16 dataset. The maximum velocity multiplier was held constant at 3 for the Burma14 runs and the tangential acceleration multiplier was set at 1 for the Ulysses16 runs.

Table 5.1. Mean Fitness on Burma14 After 30 Generations

Tangential Accel. Multiplier of 1				Tangential Accel. Multiplier of 5			
Centripetal Acceleration Multiplier of 1							
MAGS: 28.8857				MAGS: 25.9227			
	Mutation Probability				Mutation Probability		
Reset	0.0	0.2	0.4	Reset	0.0	0.2	0.4
0 %	29.1500	25.1301	25.5923	0 %	23.9335	21.3982	22.3289
50 %	28.5320	25.2781	25.2443	50 %	25.1053	22.7880	22.6635
100 %	37.3242	33.4735	30.0399	100 %	34.5916	27.8993	25.2695
Centripetal Acceleration Multiplier of 3							
MAGS: 13.4406				MAGS: 12.4977			
	Mutation Probability				Mutation Probability		
Reset	0.0	0.2	0.4	Reset	0.0	0.2	0.4
0 %	16.0331	13.5239	14.2561	0 %	13.7040	11.7084	12.3198
50 %	14.6232	14.6574	14.9645	50 %	12.5939	12.0799	12.7989
100 %	16.7505	16.0235	15.9809	100 %	14.0503	14.4526	14.0132
Centripetal Acceleration Multiplier of 5							
MAGS: 11.2927				MAGS: 9.5115			
	Mutation Probability				Mutation Probability		
Reset	0.0	0.2	0.4	Reset	0.0	0.2	0.4
0 %	12.0394	10.5117	10.9950	0 %	10.0357	8.9558	9.1681
50 %	11.3839	10.8159	10.9063	50 %	9.6649	9.2247	9.3245
100 %	11.7416	11.7746	11.3086	100 %	9.8776	9.7395	9.7876

Table 5.2. Fitness Variance on Burma14 After 30 Generations

Tangential Accel. Multiplier of 1				Tangential Accel. Multiplier of 5			
Centripetal Acceleration Multiplier of 1							
MAGS: 4.5746				MAGS: 5.2784			
	Mutation Probability				Mutation Probability		
Reset	0.0	0.2	0.4	Reset	0.0	0.2	0.4
0 %	6.1391	0.7554	1.9196	0 %	12.0456	0.5549	1.9587
50 %	8.6303	0.4765	1.4311	50 %	4.2942	3.4562	2.0219
100 %	6.3428	0.3279	10.5594	100 %	30.3444	4.4369	8.2680
Centripetal Acceleration Multiplier of 3							
MAGS: 0.2501				MAGS: 1.2824			
	Mutation Probability				Mutation Probability		
Reset	0.0	0.2	0.4	Reset	0.0	0.2	0.4
0 %	2.5257	0.5856	1.3313	0 %	2.4484	0.2743	0.3765
50 %	1.7357	1.0502	1.2457	50 %	1.5692	0.9578	1.2708
100 %	1.0354	2.3107	2.1181	100 %	1.1489	0.0097	0.2952
Centripetal Acceleration Multiplier of 5							
MAGS: 0.5566				MAGS: 0.4943			
	Mutation Probability				Mutation Probability		
Reset	0.0	0.2	0.4	Reset	0.0	0.2	0.4
0 %	1.3203	0.1146	0.4771	0 %	0.0939	0.1328	0.2615
50 %	0.2729	0.1761	0.2620	50 %	0.1885	0.3410	0.1967
100 %	0.2725	0.1631	0.5539	100 %	0.0247	0.1937	0.0921

Figures 5.1-5.3 contain a comparison of the performance of each algorithm on the Burma14 dataset using a tangential acceleration multiplier of 1 and a maximum velocity multiplier of 3 with centripetal acceleration multipliers of 1, 3, and 5. Only the best configuration of the Dubins and E-MAGS algorithms are displayed. The fitness is measured by the traversal time negative so that higher fitness is better. The green line represents the traversal time for the optimal Euclidean path using the specified vehicle limitations. Figure 5.4 compares the algorithm performances using a tangential acceleration scaling factor of 5. Displayed in Figs. 5.5-5.7 are performance comparisons on the Ulysses16 dataset for different centripetal acceleration scaling

Table 5.3. Mean Fitness on Ulysses16 After 30 Generations

Max. Velocity Multiplier of 3				Max. Velocity Multiplier of 5			
Centripetal Acceleration Multiplier of 1							
MAGS: 27.5012				MAGS: 25.0365			
	Mutation Probability				Mutation Probability		
Reset	0.0	0.2	0.4	Reset	0.0	0.2	0.4
0 %	26.8026	23.4612	24.4096	0 %	25.5275	23.4662	24.5446
50 %	28.4633	26.9928	27.4654	50 %	27.3522	26.9668	26.6572
100 %	28.3357	28.3153	27.4960	100 %	27.2023	26.5416	27.1071
Centripetal Acceleration Multiplier of 3							
MAGS: 14.5308				MAGS: 15.1523			
	Mutation Probability				Mutation Probability		
Reset	0.0	0.2	0.4	Reset	0.0	0.2	0.4
0 %	15.0196	13.4629	13.9514	0 %	14.6839	13.3649	13.9839
50 %	16.1360	15.9521	15.2794	50 %	15.6302	15.4272	15.5500
100 %	15.4155	15.8808	15.9385	100 %	16.4865	15.3848	15.3630
Centripetal Acceleration Multiplier of 5							
MAGS: 11.9803				MAGS: 11.3451			
	Mutation Probability				Mutation Probability		
Reset	0.0	0.2	0.4	Reset	0.0	0.2	0.4
0 %	11.1772	10.3550	10.8541	0 %	11.1412	10.5019	10.9220
50 %	12.4597	12.3823	12.1493	50 %	12.3073	12.2016	11.9507
100 %	12.2336	12.4309	12.3524	100 %	12.6997	12.3193	11.7966

factors while using a tangential acceleration factor of 1 and a maximum velocity factor of 3. Figures 5.8-5.10 contain performance comparisons for the same centripetal and tangential accelerations with the maximum velocity factor increased to 5.

An example of a MAGS generated path on the Burma14 dataset is shown in Fig. 5.11 and a MAGS generated path on the Ulysses16 dataset is displayed in Fig. 5.12. The blue arrows indicate the direction of travel through each node. The nodes are located at the base of each arrow.

Table 5.4. Fitness Variance on Ulysses16 After 30 Generations

Max Velocity Multiplier of 3				Max Velocity Multiplier of 5			
Centripetal Acceleration Multiplier of 1							
MAGS: 1.5720				MAGS: 1.4451			
	Mutation Probability				Mutation Probability		
Reset	0.0	0.2	0.4	Reset	0.0	0.2	0.4
0 %	3.2723	0.2416	0.8284	0 %	1.6768	0.1445	0.3556
50 %	0.8456	2.3746	0.6962	50 %	1.5829	2.9042	1.4792
100 %	0.5410	0.1858	1.2782	100 %	2.3692	1.9648	0.8353
Centripetal Acceleration Multiplier of 3							
MAGS: 0.7440				MAGS: 1.0167			
	Mutation Probability				Mutation Probability		
Reset	0.0	0.2	0.4	Reset	0.0	0.2	0.4
0 %	0.6559	0.0302	0.0968	0 %	1.1459	0.0365	0.1060
50 %	0.2466	0.1286	0.3907	50 %	0.1643	0.3809	0.1727
100 %	0.7075	0.3981	0.4364	100 %	0.6260	0.3528	0.5373
Centripetal Acceleration Multiplier of 5							
MAGS: 0.7937				MAGS: 0.2053			
	Mutation Probability				Mutation Probability		
Reset	0.0	0.2	0.4	Reset	0.0	0.2	0.4
0 %	0.7605	0.0224	0.1507	0 %	0.8230	0.0336	0.0320
50 %	0.5492	0.2383	0.1500	50 %	0.2096	0.1307	0.3916
100 %	0.1914	0.2266	0.0675	100 %	0.2295	0.5465	0.3929

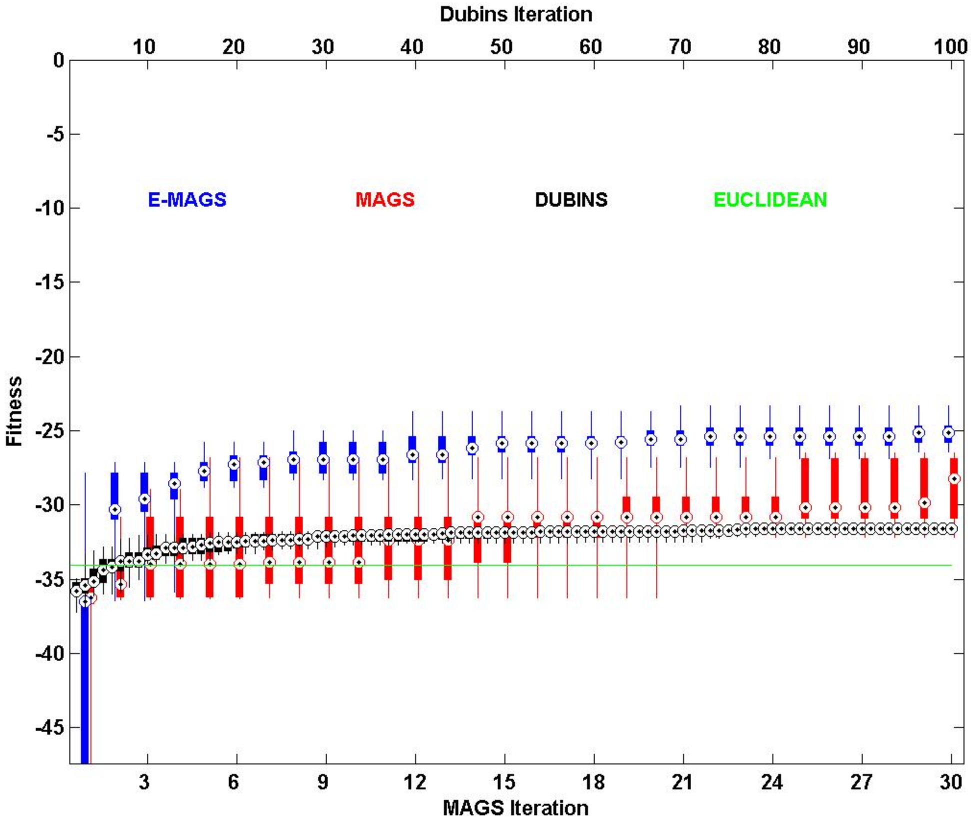


Figure 5.1. Performance Comparison on Burma14 for Centripetal Acceleration Factor of 1, Tangential Acceleration Factor of 1, and Maximum Velocity Factor of 3

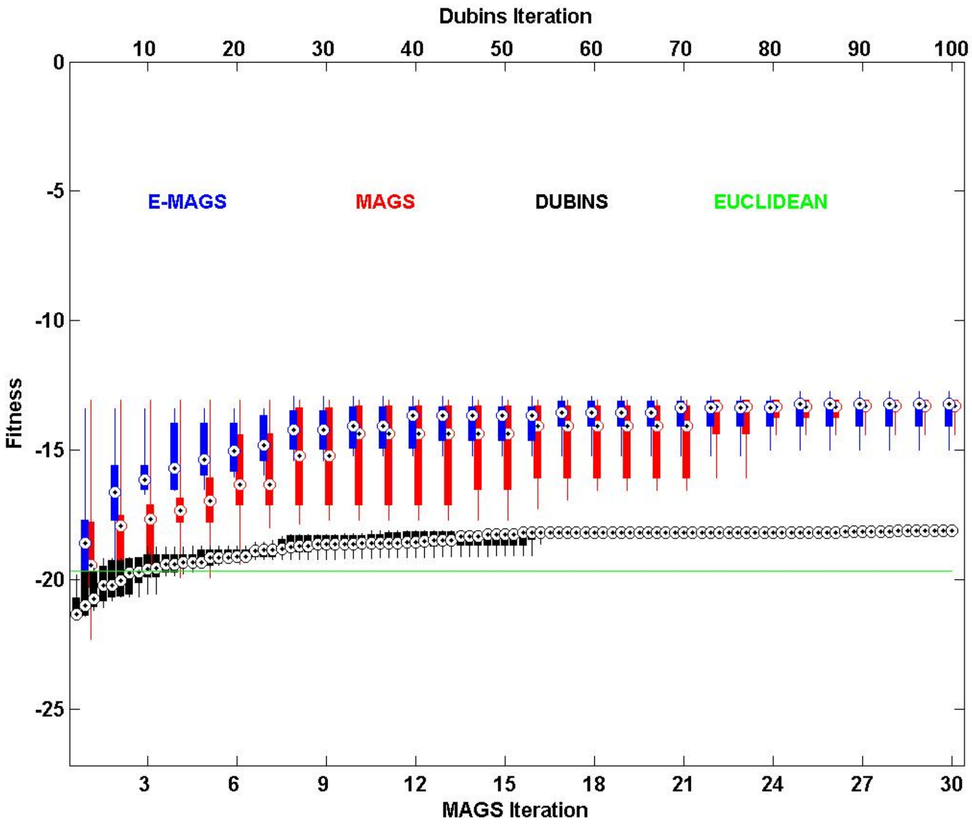


Figure 5.2. Performance Comparison on Burma14 for Centripetal Acceleration Factor of 3, Tangential Acceleration Factor of 1, and Maximum Velocity Factor of 3

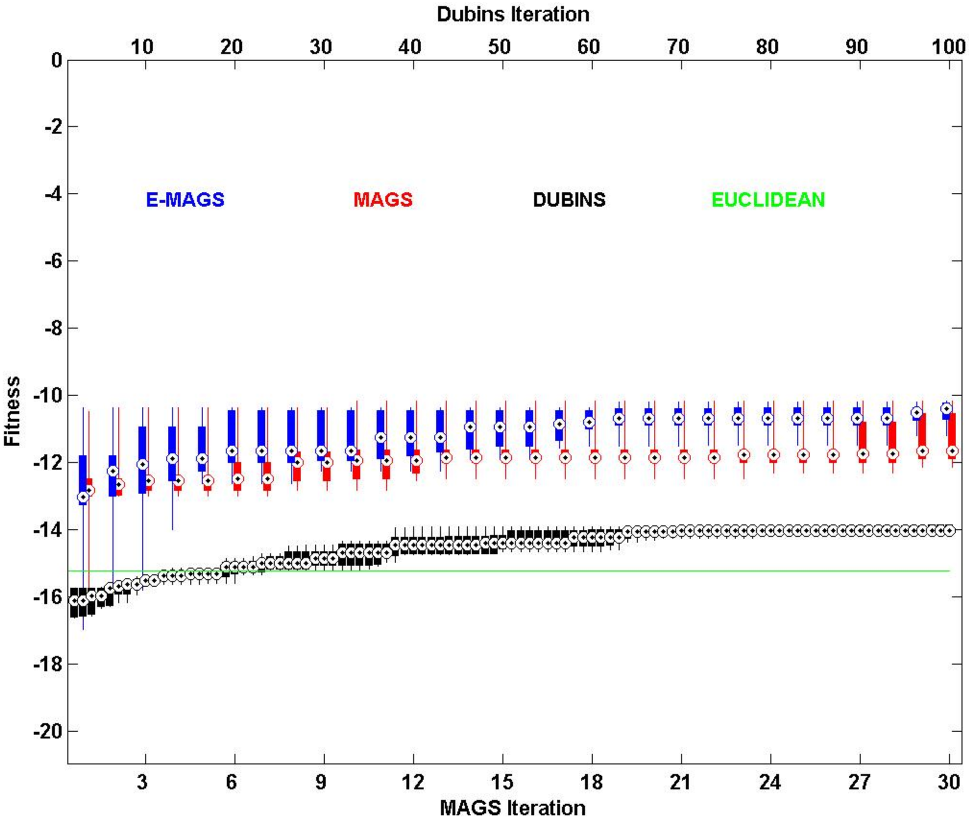


Figure 5.3. Performance Comparison on Burma14 for Centripetal Acceleration Factor of 5, Tangential Acceleration Factor of 1, and Maximum Velocity Factor of 3

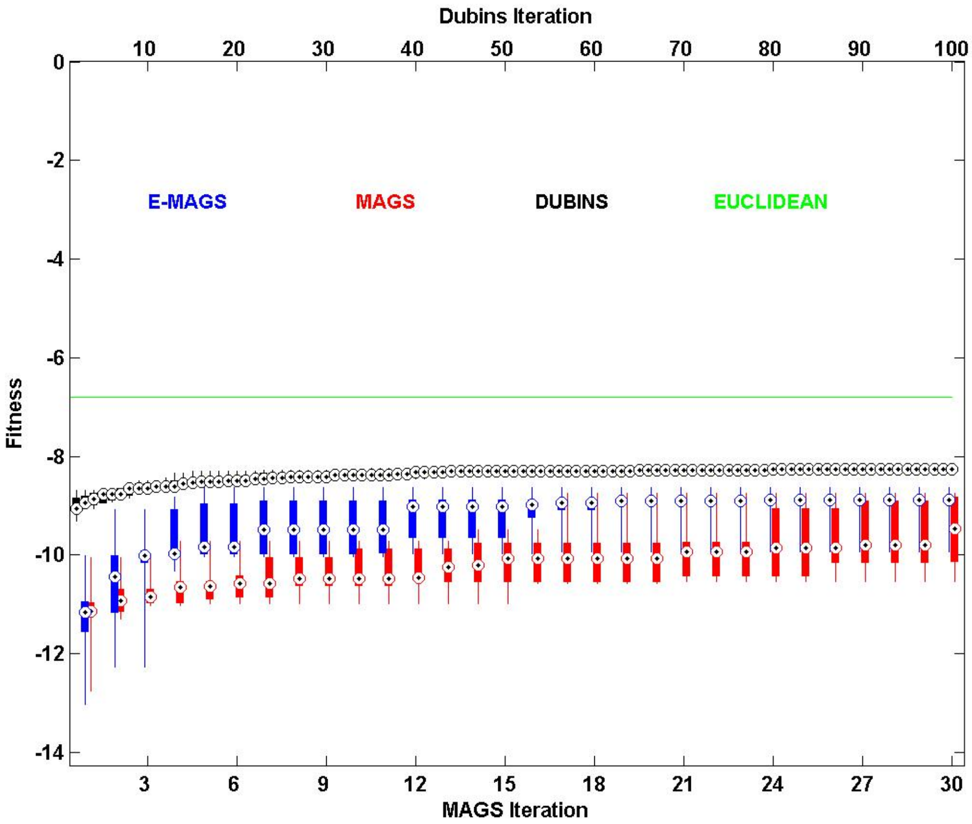


Figure 5.4. Performance Comparison on Burma14 for Centripetal Acceleration Factor of 5, Tangential Acceleration Factor of 5, and Maximum Velocity Factor of 3

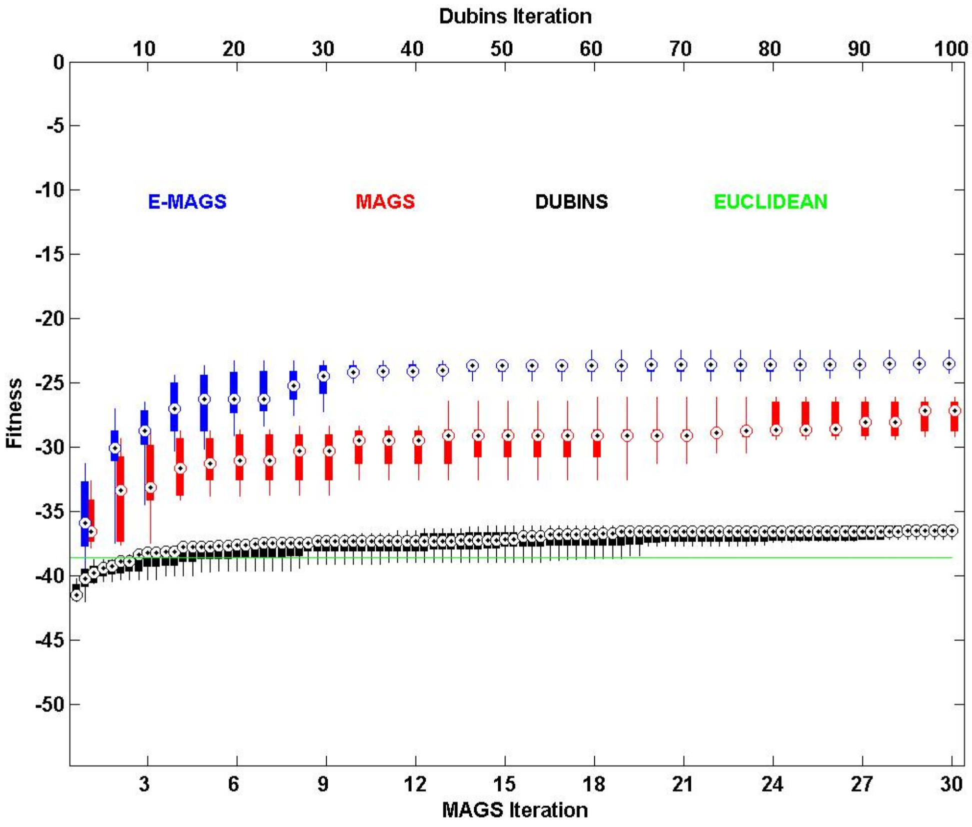


Figure 5.5. Performance Comparison on Ulysses16 for Centripetal Acceleration Factor of 1, Tangential Acceleration Factor of 1, and Maximum Velocity Factor of 3

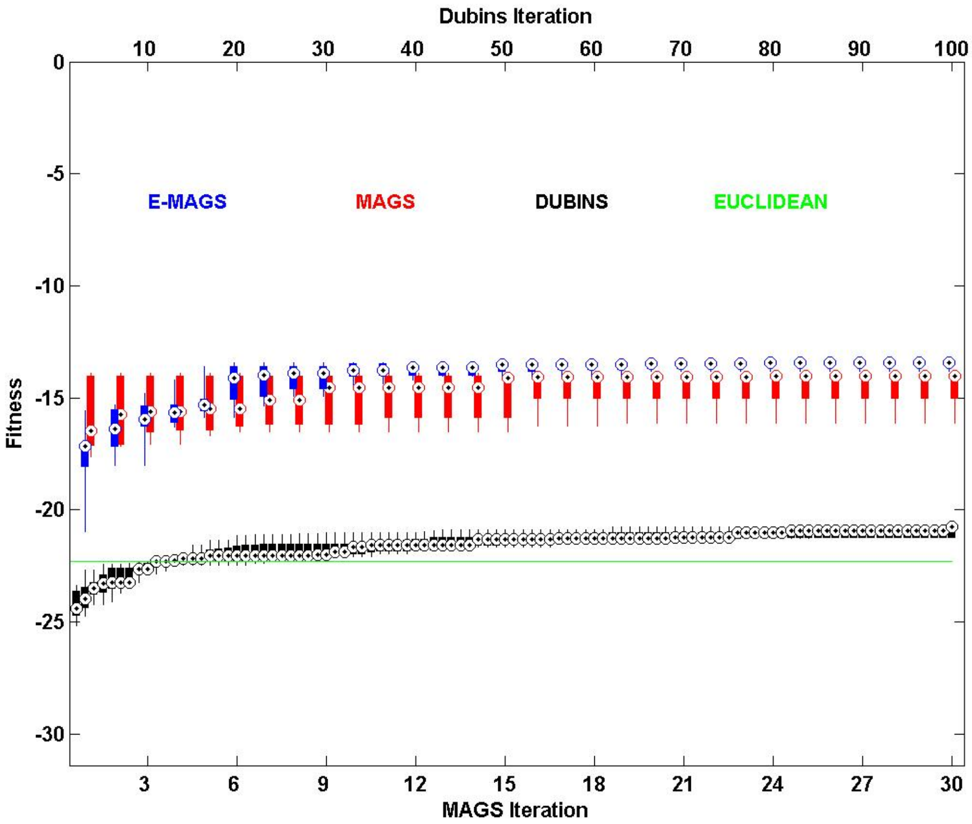


Figure 5.6. Performance Comparison on Ulysses16 for Centripetal Acceleration Factor of 3, Tangential Acceleration Factor of 1, and Maximum Velocity Factor of 3

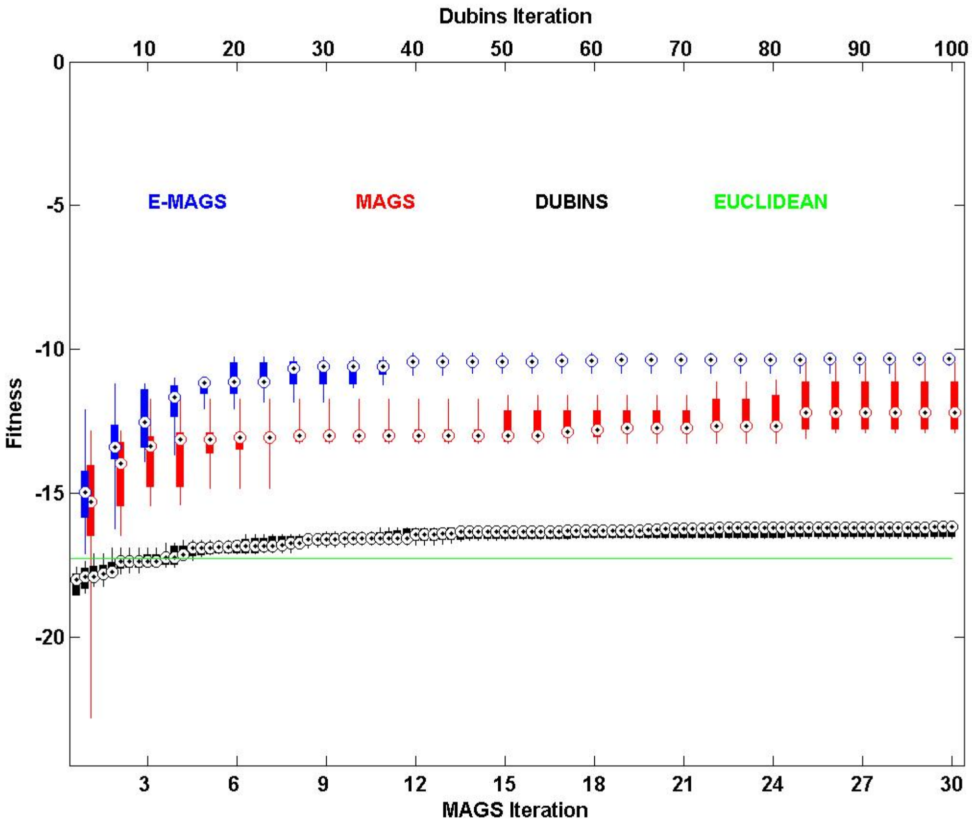


Figure 5.7. Performance Comparison on Ulysses16 for Centripetal Acceleration Factor of 5, Tangential Acceleration Factor of 1, and Maximum Velocity Factor of 3

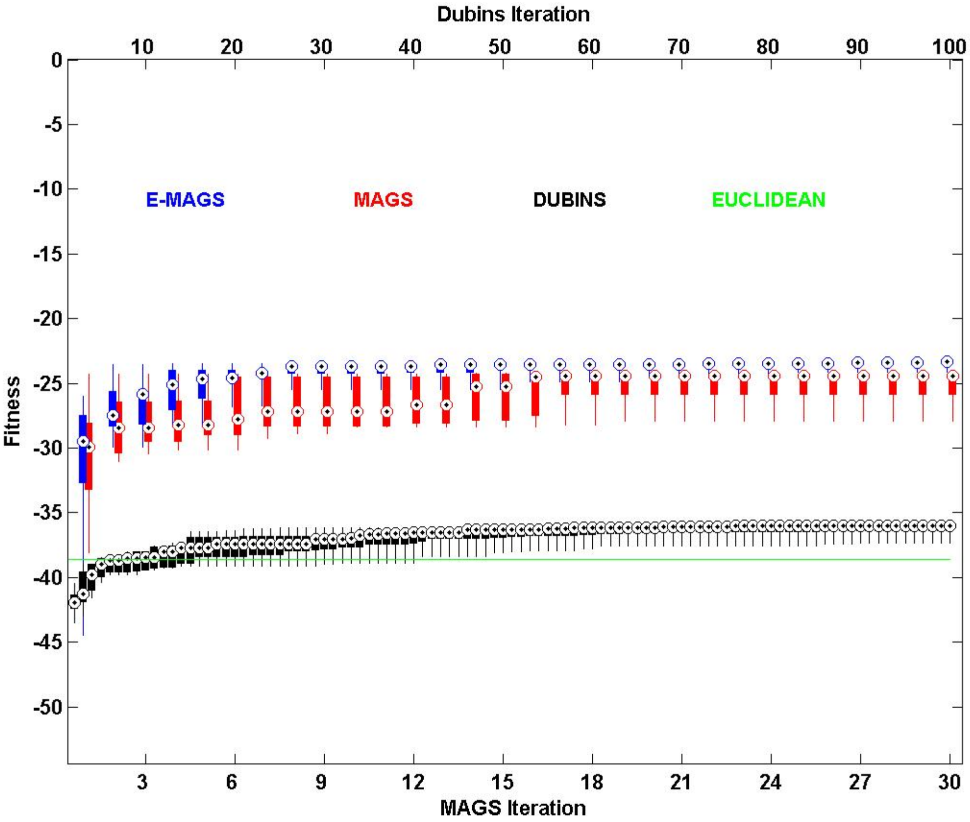


Figure 5.8. Performance Comparison on Ulysses16 for Centripetal Acceleration Factor of 1, Tangential Acceleration Factor of 1, and Maximum Velocity Factor of 5

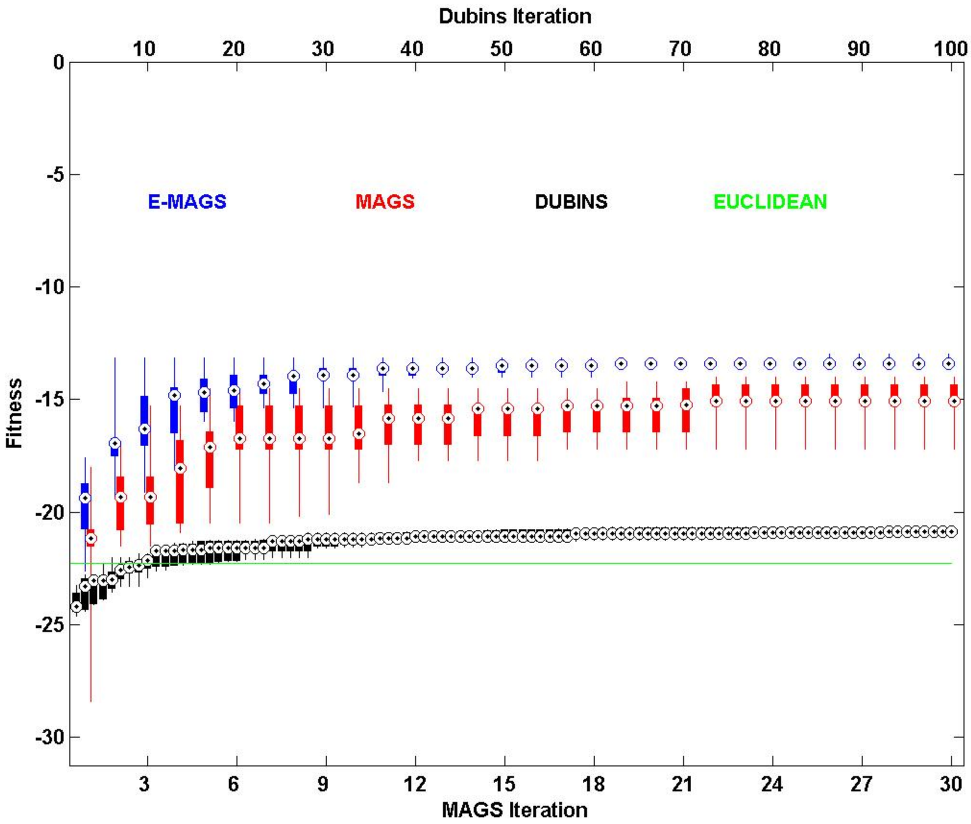


Figure 5.9. Performance Comparison on Ulysses16 for Centripetal Acceleration Factor of 3, Tangential Acceleration Factor of 1, and Maximum Velocity Factor of 5

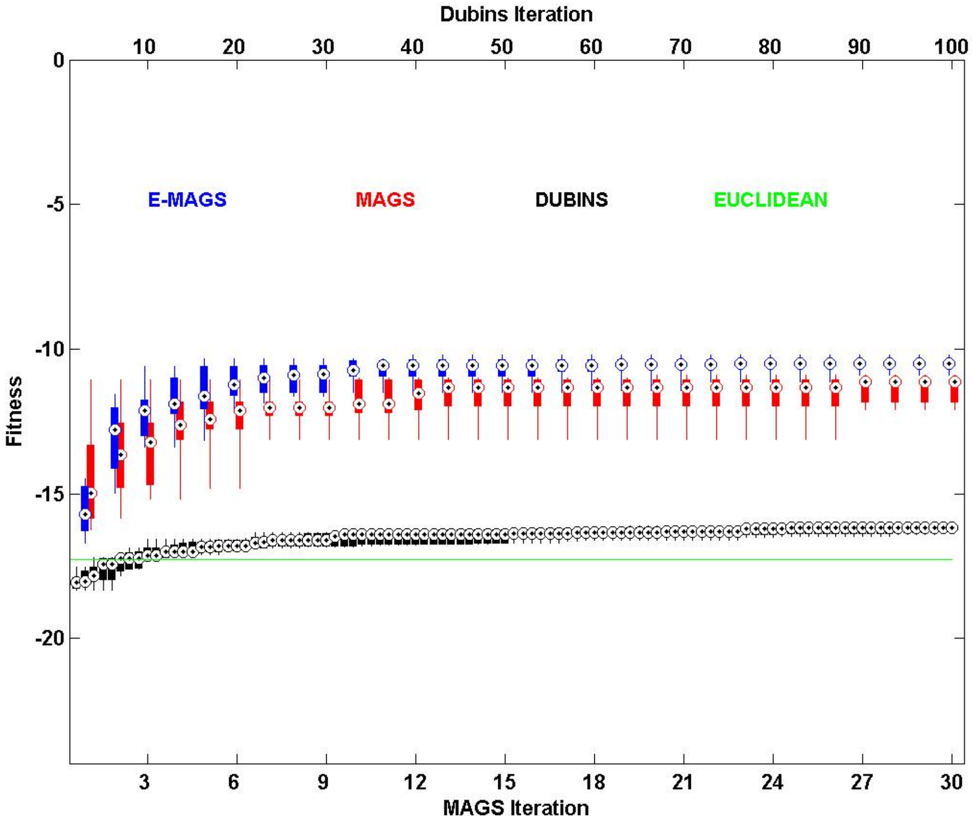


Figure 5.10. Performance Comparison on Ulysses16 for Centripetal Acceleration Factor of 5, Tangential Acceleration Factor of 1, and Maximum Velocity Factor of 5

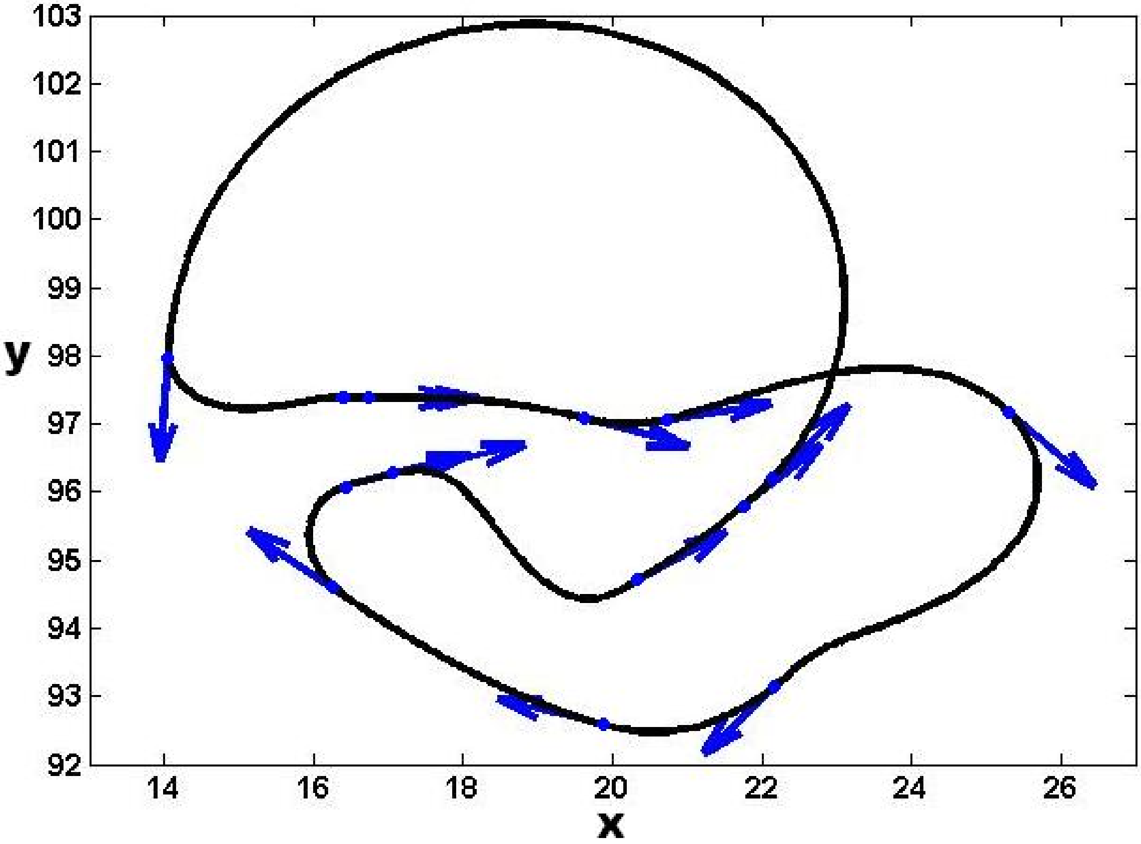


Figure 5.11. Example MAGS path for the Burma14 dataset

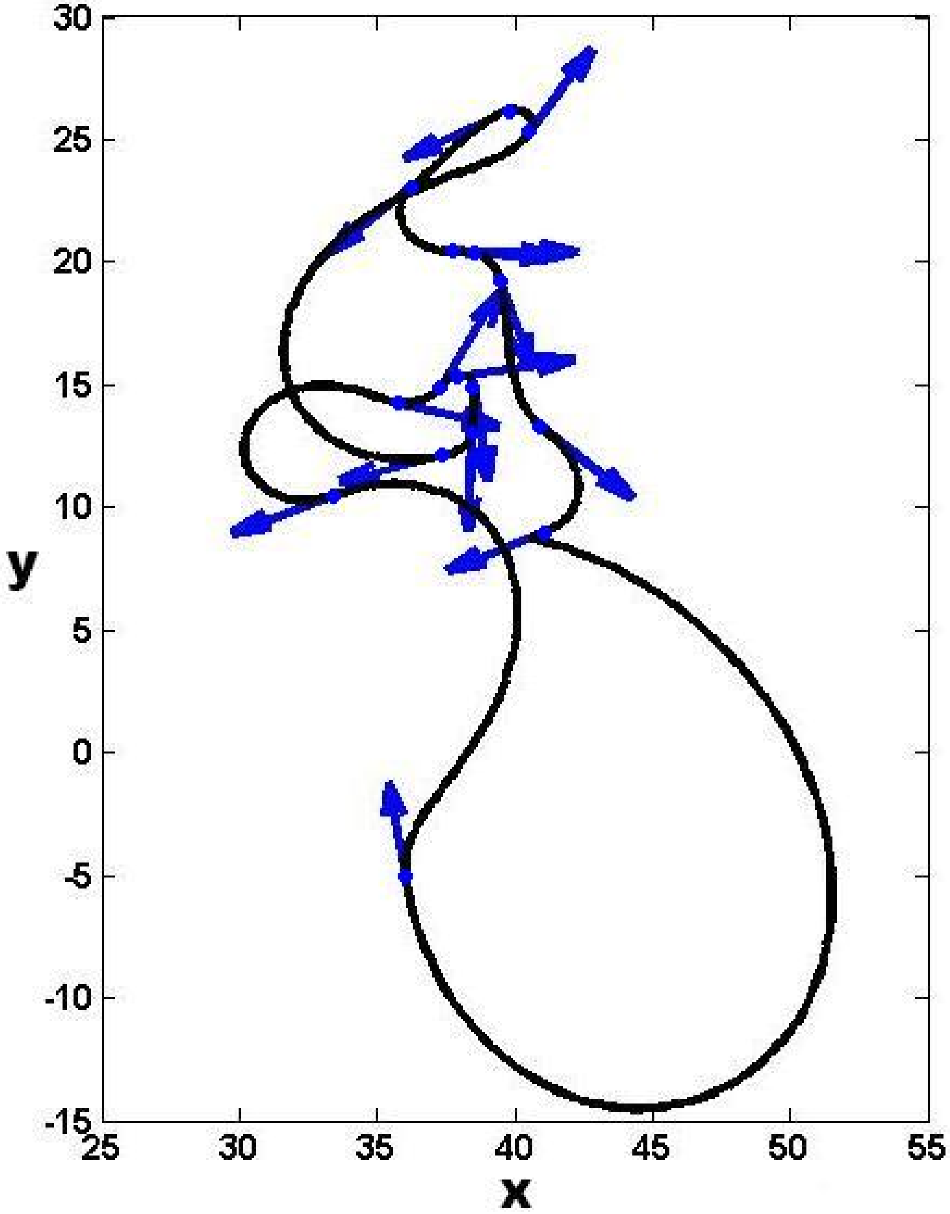


Figure 5.12. Example MAGS path for the Ulysses16 dataset

6. DISCUSSION

An examination of the performance of the evolutionary operators tested reveals that the best performance for all vehicle specifications tested was always obtained with no reset of the bond connections and a 20% mutation rate. This configuration typically also had one of the lowest variance in the data as well. It is also noted that the partial reset often performed well and in some cases using the Burma14 dataset, was better than no reset when mutation rates other than 20% were used. It is therefore recommended that the bond connection reset should be minimized, but a small percentage of reset may be beneficial if the mutation rate is not optimized. The reset percentage had a larger effect on the performance than that of the mutation rate, revealing that the ordering information encoded within the bonds is necessary for optimal performance of MAGS. This is likely because when nodes are fully connected, nodes can be affected by the orientation of nearby nodes even if they are distantly connected in path ordering. This can lead the nodes to be influenced away from optimal configurations. The mutation rate was discovered to be of significance, though its effects were not as large as the bond reset. Performance was observed to decrease with both too large and too small mutation rates. In all cases except for one, the best E-MAGS fitness exceeded that of MAGS. Non-optimal E-MAGS configurations also yielded comparable results to MAGS although in many cases inferior.

In comparing both MAGS and E-MAGS with the Euclidean and Dubins paths, significant improvements were seen in all cases except for when a tangential acceleration scaling of 5 was used. In this case, the Euclidean path became the optimal choice since the need to slow to a stop at each node was not a significant penalty in comparison to the path length. In cases such as these, the Euclidean path is the best choice unless the vehicle has mechanical curvature constraints that limit the sharpness of the turns.

A visual inspection of the paths yields further insight into the performance. In both Fig. 5.11 and Fig. 5.12, the ordering of the nodes is dramatically different from the optimal Euclidean ordering, and allows larger curvature radii to be maintained. However, the path in Fig. 5.12 does illustrate some of the inherent weaknesses of the curve generation method used. The curve connecting the lowest two points contains an unnecessarily sharp turn. This problem was often encountered in cases where the angles of the two nodes are greatly misaligned because of the influence of other nodes. If the algorithm were run long enough, this might correct itself through adjustment in the node angles, but it would be desirable to have a better trajectory generation for cases when misalignment between two nodes persists.

7. CONCLUSIONS

Current published methods for vehicle trajectory planning for TSP problems typically focus on solving curvature constrained TSP and are primarily limited to planning for constant speed travel using the Dubins vehicle model. These methods have been shown to work well for widely spaced targets, and for vehicles that travel with constant speed. However, when targets are clustered more densely, these paths can become increasingly sub-optimal for vehicles that have the ability to vary their speed since these methods focus on minimizing path length rather than traversal time.

MAGS shows great promise in quickly converging on a local optimum solution. The solution is based on a balance between minimizing path length and curvature rather than limiting the vehicle to a single speed. The primary drawback to MAGS is that it locates a solution deterministically based on the initial configuration. This causes it to be easily trapped by local minima. The evolutionary hybridization proposed in this research significantly improved the performance of the MAGS algorithm by encouraging exploration. Various evolutionary operators were compared to determine the effectiveness of each. The recombination of the connection weights was shown to be much more effective than letting each individual start fresh with the new angles. The mutation rate was found to have some effect in the number of generations observed, with low mutation rates being favored in early generations, but with higher mutation rates leading to long-term improvement.

It was also shown that allowing MAGS to start with full connection weights causes interference between nearby nodes and draws the solution to inefficient local minima traps, preventing optimal performance. By using evolutionary operators to choose preferred node connections for the initialization of MAGS the performance was improved.

The computation time necessary for each generation currently makes large problem sizes or large evolutionary populations impractical. The MAGS code is still in development, however, and has not yet been fully optimized for efficiency. Further improvements may also be achieved by implementing portions of the code with a GPU in order to take advantage of the parallel nature of much of the computation required for MAGS. Much of the computation necessary for the MAGS algorithm is highly parallel in nature. Therefore, the computation time necessary could likely be reduced significantly by utilizing a GPU to run parallel calculations whenever possible. The quality of the MAGS results could also possibly be improved by implementing alternate methods of determining where to merge multiple loops that naturally form.

The most significant difficulty still present in achieving optimal results with MAGS and E-MAGS and likely the greatest opportunity for improvement lies in generating optimal curves between nodes for which efficient curvature analysis is possible. Some error due to performing discrete approximations of the inverse velocity integral were observed, particularly in cases where the velocity was close to zero. Additionally, the curve generation method used in this study works well when node angles are nearly congruent, however in some cases the curve generation method resulted in sub-optimal path segments when nodes were widely separated or when node angles were misaligned.

As continuing advancements are made in computation power and efficiency, path planning methods that focus on traversal time fitness rather than path length will be necessary and practical. This shift in focus leads to some unique challenges that are not present in traditional TSP or even many curvature-constrained TSP methods. MAGS and E-MAGS have shown promise in providing a way to meet these challenges and offer several opportunities for further improvement and exploration within this field.

BIBLIOGRAPHY

[1] L. E. Dubins. On Curves of Minimal Length With a Constraint on Average Curvature and with Prescribed Initial and Terminal Positions and Tangents. *American Journal of Mathematics*, 79(3):497–516, 1957.

[2] R. Kenefic. Finding good dubins tours for uavs using particle swarm optimization. *Journal of Aerospace Computing, Information and Communication*, 5(2):47–56, February 2008.

[3] J. Kennedy and R. Eberhart. Particle swarm optimization. In *IEEE International Conference on Neural Networks*, volume 4, pages 1942–1948, 1995.

[4] J. Le Ny, E. Frazzoli, and E. Feron. The Curvature-constrained Traveling Salesman Problem for High Point Densities. In *Decision and Control, 46th IEEE Conference on*, pages 5985 –5990, Dec. 2007.

[5] C. Lim, S. Park, C. Ryoo, K. Choi, and J. Cho. A Path Planning Algorithm for Surveillance UAVs with Timing Mission Constrains. In *Control Automation and Systems, International Conference on*, pages 2371–2375, oct 2010.

[6] X. Ma and D. Castanon. Receding Horizon Planning for Dubins Traveling Salesman Problems. In *Decision and Control, 45th IEEE Conference on*, pages 5453–5458, dec 2006.

[7] K. May, H. Sien, Y. Ping, and S. Z. Hai. An Evolutionary Algorithm for Multiple Waypoints Planning With B-spline Trajectory Generation for Unmanned Aerial Vehicles (UAVs). In *2010 International Conference on Computational Problem Solving*, pages 77–81, 2010.

[8] C. H. Papadimitriou. The Euclidean Travelling Salesman Problem is NP-complete. *Theoretical Computer Science*, 4(3):237 – 244, 1977.

[9] G. Reinelt. http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/, June 2011.

[10] K. Savla, F. Bullo, and E. Frazzoli. On Traveling Salesperson Problems for Dubins Vehicle: Stochastic and Dynamic Environments. In *Decision and Control, 44th IEEE Conference on*, pages 4530 – 4535, Dec. 2005.

[11] G. Yang and V. Kapila. Optimal path planning for unmanned air vehicles with kinematic and tactical constraints. In *IEEE Conference on Decision and Control, Procceedings of*, volume 2, pages 1301–1306, 2002.

VITA

Jared Adam Nisbett was born in 1987 in Arlington, Texas, USA. He received a B.S. in Mechanical Engineering from Missouri University of Science and Technology in May 2008. As an undergraduate, Adam received the Academy of Mechanical and Aerospace Engineering Student Excellence Award. Adam was active in Pi Tau Sigma, the Mechanical Engineering Honor Society, serving as Website Chair for several semesters. As a graduate student, Adam was awarded the Chancellor's Fellowship. Adam placed third in both the 2009 Intelligent Systems Center Research Symposium as well as the 2010 Intelligent Systems Center Poster Presentation. Adam has also been actively involved in various community activities, mentoring high school robotics teams, teaching art and chemistry classes for home schooled students, as well as designing set and performing with local community theatre.