A Primate-inspired Autonomous Navigation Algorithm Using the Cognitive Mechanism of Mental Rotation

Michael J. Pettinati and Ronald C. Arkin

Georgia Institute of Technology, College of Computing, Atlanta, GA, 30332, USA
mpettinati3@gatech.edu, arkin@cc.gatech.edu

Abstract. Though significant progress on autonomous navigation has been made, the natural world offers interesting examples of navigational techniques that are worth exploring and understanding. The cognitive mechanism of mental rotation has been revealed in numerous cognitive and neuroscientific experiments; its reason for existence and evolution, however, has yet to be thoroughly understood. It is speculated that this mechanism may assist primates in navigation. This paper explores how mental rotation can be used in navigation by developing an autonomous robotic navigation algorithm that draws inspiration from the mechanism. This algorithm was tested on a robot tasked with navigating to a specified goal location contained within the agent's initial view. The testing suggests that mental rotation can be used as an asset in navigation.

Keywords: mental rotation, robotic navigation, mental imagery

1 Introduction

Autonomous navigation is a problem with a set of robust and efficient solutions [1,2]. The fact that these solutions are sufficient for many applications does not imply that autonomous navigation is a domain where no further progress needs to be made. The natural world offers interesting examples of alternative navigational techniques that are worth taking the time to understand and may offer insights that supplement or enhance existing algorithms.

Evolution has fashioned primates into adept and efficient navigators [3,4]. The ability of nonhuman primates (and human children) to use mapping is limited, therefore, other mechanisms must have evolved in primates to allow for successful navigation [5]. Many studies have identified the mechanism of mental rotation, an ability that allows primates to "envision" a reorientation of an object/scene [6,7,8]. There has been speculation that this mechanism contributes in some way to the navigational capabilities of primates, but there have been few studies conducted to verify these speculations.

The goal of this ONR-funded project is to understand what mental rotations might contribute to the ability of primates to navigate by implementing an autonomous navigation algorithm that draws inspiration from the ability of mental rotations, incorporating this algorithm into our existing robotic specification software, *MissionLab* [9]. In theory, given the view of a scene from a goal location and a view of the scene from the robot's current location, an agent can use "mental" rotations to visualize the current view at a three-dimensional orientation that is aligned with its goal's three-dimensional orientation. This view will allow the agent to accurately assess the translational and rotational offset between the current position and goal position.

It is posited that this research could provide the foundation for a navigation algorithm that supports advice giving in robotic navigation. If an agent did not "know" its goal view a priori, something/someone external to the agent could provide a high level description of its goal by relating its relative position and orientation to known objects in the scene. The agent could then use these objects to define the scene and travel appropriately using the biologically-inspired algorithm introduced below. For example, if an agent had to fetch something from a file cabinet in an office within an office building, it might have a map to the office, but it might not know the precise layout of every office in the building. If the agent knew what a file cabinet looks like, upon arriving at the office, it could identify the file system (if there are multiple, one could be specified to the agent) and navigate to the appropriate position relative to the file cabinet using an algorithm like the one described here to carry out its task.

2 Related Work

Research has shown that although human children and nonhuman primates are robust navigators, they do not use maps or precise distances to navigate to a location [3,5]. It is suggested that they instead use mental transformations to overcome changes in perspective and to make general assessments about the direction in which they must move to reach a certain location. Research by Kosslyn et. al. [10] affords more concrete evidence for the use of mental images in humans through neuroimaging studies.

Hutcheson and Wedell [11] discuss how, when humans are immersed in an environment and navigating to a certain location, they tend to use these qualitative, abstract representations to navigate as opposed to using more precise distances or explicit directions; Menzel et al. [4] observed a similar tactic with a nonhuman primate. The Bonobo being observed had to travel from a start location to a designated goal; it did not take a rigid trajectory but varied its path [4]. Starting position did not affect its ability to successfully navigate, implying it possesses the ability to mentally encode the "entire spatial layout" of the area, and can mentally manipulate this encoding to localize itself and travel appropriately [5].

Mental rotation must be differentiated from the cognitive process of perspective-taking spatial abilities and visual servoing or homing techniques. Hegarty and Waller [12] show an explicit distinction between cases where an observer is mentally manipulating a scene or scene object to view it differently and cases where the observer is mentally viewing the scene from a different viewpoint. They [12] state that humans may use both of these skills, but most people have a strong preference for one or the other. As expressed by Arkin [13], visual servoing is distinct from the navigational approach described here in that the visual approach described in this paper is a more deliberative approach. Our approach, described in more detail below, will derive the appropriate navigation direction using abstract representations of the scene rather than working with image features. The three-dimensional, structured representations of the scenes used by our algorithm allow for correspondence between elements of like form in the different scenes (models of identifiable objects or object parts) as opposed to corresponding image features. In the context of advice-giving, full object recognition and semantic labeling will be required.

The evidence presented above has made a case that primates can maintain abstract mental images and manipulate these representations. The navigation approach introduced below has an agent rotating a "mental" visual representation of an object in order to inform its motion. It has also been theorized that mental rotation could be accomplished through the manipulation of propositional logic statements [13], which has received less support than the visual analog approach. We use the visual analog approach.

Stein [14] presents the idea of a robotic system using "mental" simulation in order to guide itself toward a goal location. Stein's [14] system, MetaToto, used an extended simulation where the agent would "imagine" what its sensors would experience at various locations on a floor plan and build a world model. This project is not as deliberative as MetaToto. This work explores a biologically-inspired, semi-reactive navigation algorithm that makes use of a process inspired by mental rotation to continually inform/update the movement of the robotic agent.

3 Navigation Algorithm

3.1 Algorithm Formation

This lab has previously developed a navigational procedure that explored the mechanism of mental rotation [13,15,16]. This system used depth information in order to construct occupancy grids that represented the current view of the scene and the goal view of the scene. Figure 1 shows a view for the robotic agent, the depth image capturing that view, as well as the occupancy grid that was generated from that depth information. The points composing the generated occupancy grids had discrete states of occupied or unoccupied, projecting the scene onto the two dimensional depth and width space.

At the outset of its mission, the agent generated the occupancy grid given the depth image for both its goal and its current location. The agent used a discrete number of "mental" transformations on the current occupancy grid in an attempt to match it to the goal occupancy grid. The correspondence between the two occupancy grids was scored for each transformation. A motion vector was derived from the transformation that resulted in the closest correspondence between the two occupancy grids [13,15]. After the robot had moved a short, specified distance, it would capture the depth information at its new current location, generate the occupancy grid representation at this new location, and repeat the process until the occupancy grids sufficiently corresponded without requiring "mental" transformations[1]. This algorithm had the depth imagery projected onto the ground plane, which is not biologically plausible. To be biologically plausible, the algorithm must work directly within the view of the scene [16].

The neuroscience literature discussing mental rotation is often focused on the rotation of simple objects, not entire scenes. The work done by Aretz and Wickens [6] and by Bläsing et. al. [7] each note a certain amount of fragility in the mechanism of mental rotation. The work presented by Khooshabeh and Hegarty [8] found that in order to overcome this fragility many humans will segment a scene/complex object into distinct parts and mentally rotate these parts individually. This paper's navigation algorithm begins by segmenting the view into discrete planar segments that can be acted on individually.

3.2 Segmentation Algorithm

An overview of the segmentation algorithm used in navigation is presented here; for more depth on its theory, see Erdogan et. al. [17] and Srinivasan and Dellaert [18]. The segmentation algorithm takes as its input RGBD images; these images are captured via the Microsoft Kinect. To begin, these images are smoothed using a bilateral, edge-preserving filter. In order to decrease the size of the algorithm's search space when labeling planar segments, the pixels composing the images are grouped into related, atomic regions known as superpixels. Pixels are uniquely assigned to a superpixel; the pixels are grouped into superpixels based on three different factors: spatial proximity, color, and disparity. The result of performing this "oversegmentation" on the goal views of the two different scenes introduced below are shown as part of Figure 2. The noisiness of the oversegmentations seen is largely due to the limited smoothing of the input RGBD images from the Kinect. The neighborhood in which the bilateral filter smoothed was restricted to allow for the agent's direction to be updated more regularly. As long as the segment(s) the agent is using for navigation is/are consistently recovered, the surrounding noise can be ignored successfully.

Once the oversegmentation has completed, the superpixels are grouped probabilistically using the Rao-Blackwellized Markov Chain Monte Carlo (MCMC) algorithm presented by Srinivasan and Dellaert [18]. Each segmentation consists of a set of planar segments where each planar segment is a set of superpixels. There are eight hundred segmentations generated by the algorithm, and the most commonly occurring segmentation is returned as the three-dimensional representation of the view. The chosen segmentations for the two tested goal views are shown in Figure 2.

---

[1] A video demonstrating these results is available at: `http://www.cc.gatech.edu/ai/robot-lab/brc/movies/brc_video.mp4`

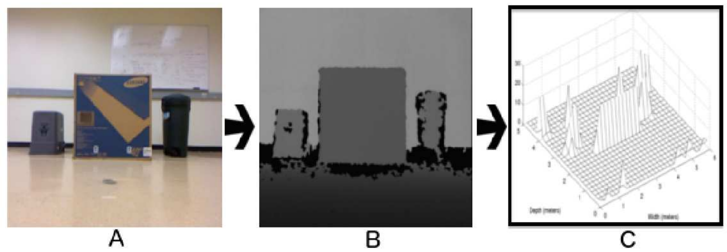

Fig. 1. A) The image of the scene. B) The depth image captured at that view. C) The occupancy grid generated from the depth information.

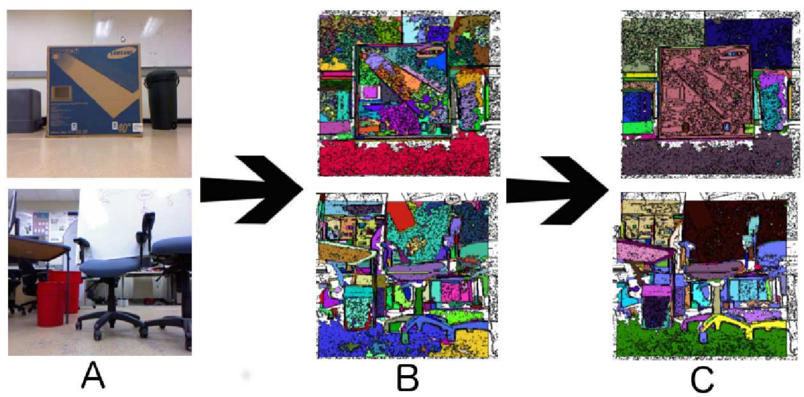

Fig. 2. A) View of scene at goal location for both tested scenes. B) Oversegmentation of goal image for both tested scenes. C) Final segmentation of goal image for both tested scenes. The first test scene appears in the top row. The bottom row contains the second test scene.

3.3 Navigation Algorithm

A high level overview of the navigation algorithm appears in Figure 3. The algorithm is provided with a conception (RGBD image) of the goal. The algorithm immediately captures an RGBD image of its current view using an onboard Microsoft Kinect. It is able to generate a probable planar segmentation, similar to those seen in Figure 2, for both the goal view and the current view using the segmentation algorithm described above. The algorithm must now match segments in the goal view with the corresponding segments in the current view to try to align them using a process inspired by mental rotation. The two views have different segment sets and many of the segments are inconsequential to the scene; therefore, the segments that truly define the scene such as the box in the first scene's goal view or the white board in the second scene's goal view must be identified as segments that the algorithm will use to move the agent to the goal. Ultimately, this process will be automated, however, currently in this bootstrap phase a human user identifies the segments "key" to the goal view and the corresponding segments in the initial current view. After this bootstrap step is complete, the agent navigates to its target goal location autonomously.

A human hand-matching the “key” segments for the agent is a shortcoming of this algorithm; however, the bootstrap step will be a focal point of the research moving forward. In the context of advice-giving, something external to the agent will still be identifying what is “key” to the agent for navigation (though not identifying it in either scene). The agent will have to be given or generate a model of the object(s) described by the person assisting it and identify and match these objects for itself in the starting view and goal view.

Immediately following this bootstrap step, the algorithm considers each pair of corresponding segments in turn. It computes the average normal vector for the planar segment in both images. The estimated average normal vector for each plane segment is computed by estimating the normal at each pixel in the given segment using PCL’s normal estimation [19] and averaging each pixel normal in the segment. The algorithm attempts a discrete number of rotations to align the current normal vector with the goal normal vector. In a process attempting to mimic mental rotation, the normal vector of the current segment is gradually rotated around each of the three axes. After each rotation, the inner product between the current view segment’s normal vector and the goal view segment’s normal vector is computed. The maximum value of this computation corresponds to the mental rotation that most closely aligns the orientation of the current view and goal view. The algorithm estimates the offset of the segment in the current view from its position in the goal by using the mental rotation (i.e. using the rotation matrix that aligned the normal vectors) to “visualize” the segment at the same orientation as the goal and determine the three-dimensional spatial distance between the center point of the segment in the goal view and the rotated center point of the segment in the current view. The algorithm makes this assessment for each of the corresponding planes designated as “key” in the scene and finds the average rotation and average estimated offset.

Due to the limited field of view of the Microsoft Kinect, the algorithm is not going to simply send the agent in the direction of the goal. If the agent is oriented and does not simply have to move in depth to reach the goal, the algorithm recommends the agent moves left/right (whichever direction is advised by the comparison stage) as quickly (as much) as possible to avoid being turned away from the segments later in the navigation when they can be more easily lost from view. If the agent is not oriented with its goal view orientation, then the algorithm will orient the agent to avoid losing segments in the periphery of the agent’s view. These decisions assumed the agent could “see” all segments used to navigate when oriented at the start and goal locations. After making this assessment, the algorithm sends the angle that points the agent in the appropriate direction. A depth difference between the goal view and current view is also sent.

Once the robot begins to move, the system captures its new current view and uses the segmentation algorithm described above. As Arkin [20] suggests, finding the segments that correspond in the goal view and this new current view is a less complex problem than the bootstrap step. This feedforward step uses information about the robot's motion, the robot's speed and the angle just given to the system, as well as information about the positions of "key" segments in the previous current view to find a correspondence between the segments in the new current view and the goal view. The navigation algorithm considers each segment in the new current view and attempts to inversely map these segments to the previous current view based on the speed of the robot and its most recent direction to see if the segments fall within regions occupied by "key" segments in the previous current view. To limit the number of segments considered, the algorithm only considers segments that are approximately at the estimated depth of the matching segment. After the segments in the new current view that correspond to "key" segments in the goal view have been identified, the algorithm uses the process inspired by mental rotations to make an assessment about the agent's orientation and the appropriate direction in which it should move. This process is going to continue until the agent reaches the depth of the goal location. The navigation concludes at the goal depth because the agent cannot turn ninety degrees without losing the segment(s) being used for navigation.

```
Data: RBGD goal image
begin
|                         Bootstrap Step
|   capture first current view RGBD image
|   perform planar segmentation on start view and goal view
|   identify goal view segments and match them to corresponding start view segments
|   for each key segment do
|   |   estimate rotation between segment in current view and segment in goal view
|   |   get approximate offset between current view segment and goal view segment
|   end
|   compute average rotation and offset between current view segments and goal view
|   segments
|   send appropriate motion vector to robotic agent
end
begin
|                         Feedforward Step
|   while true do
|   |   capture new current image
|   |   match key segments in new current to those in goal
|   |   for each key segment do
|   |   |   estimate rotation between segment in current view and segment in goal view
|   |   |   get approximate offset between current view segment and goal view segment
|   |   end
|   |   if at goal depth then
|   |   |   send angle to orient the agent/zero goal depth
|   |   |   kill/stop the agent
|   |   end
|   |   send appropriate motion vector to robotic agent
|   end
end
```

Fig. 3. Overview of navigation algorithm.

4 Results/Analysis

The algorithm was tested on a Pioneer 2-DX robotic platform. The testing was designed to demonstrate the ability of the algorithm to guide the robotic agent from a variety of starting locations and orientations to a specified goal location and orientation. The successful navigation of a robotic agent using this algorithm would lend support to the idea that mental rotation can serve primates in navigation, and it would provide a foundation for the referenced advice-giving algorithm.

Two different scenes were tested (Figure 2A). Thirty trials were run on each scene. There were three, ten trial experiments conducted for each scene. Each of the ten trial experiments was run with the robotic agent starting at a certain location and orientation and navigating toward the goal location defined for that scene. A trial was only deemed successful if the agent was within 0.5m of the goal location, and the difference between the agent's final orientation and the goal orientation was no more than 10°. The distance from the goal location was measured from the robot center to the goal location. Due to the limited range of the Kinect, trials were restricted to situations where the agent's starting locations were 4.5m from the scene or less. Because the segmentation algorithm extracted planar segments, each trial included at least one planar element that was "key" to the scene to ensure reliable segmentation so the average normal vector would accurately represent orientation.

Figure 4 shows the finite state acceptor (FSA) defined for the robotic agent in *MissionLab*. The robot's behaviors/states are the circular symbols and the triggers for these behaviors are rectangles. In this mission, the agent stops (*MENTALALIGNED* is true) when the navigation algorithm indicates that the agent is oriented and an acceptable depth difference has been reached; the robotic agent has reached the goal location at this point. The *MoveMentalDirection* behavior is triggered whenever the agent still has to travel to reach its goal; the behavior uses the heading computed by the navigation algorithm to move in the appropriate direction toward the goal. The *OrientAtGoalDepth* behavior is triggered when the agent cannot translate without going beyond the goal yet still must orient itself properly.

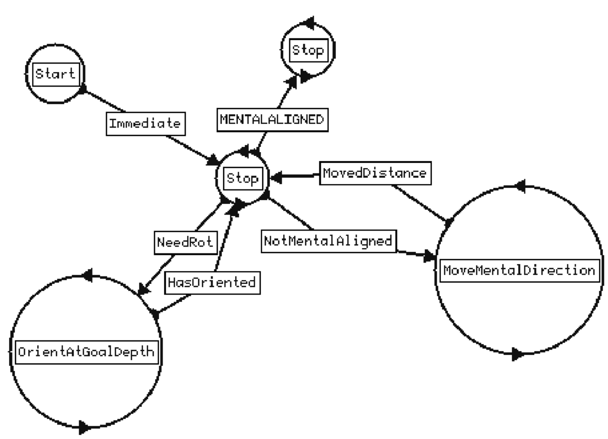

Fig. 4. FSA for robotic agent.

4.1 Original Scene Revisited

The first test scene (Figure 2A) contains 3 different objects: a television box, an overturned crate, and a trash barrel. The front of the box was identified as the "key" planar segment that the agent used to navigate in all trials. The depth of this surface was uniform, 2.0m. The agent was pointed directly at the center of the box when the goal RGBD goal image was captured. The results for the thirty trials run at this scene are summarized in Table 1 below. The horizontal displacement is negative if the robot stopped to the goal's left and positive to the right. The difference in depth from the goal was positive if the agent went beyond the goal and negative if the agent stopped before the goal. The rotational offset is positive if the agent finished oriented to the right of the goal orientation.

Table 1. Results: Scene 1. Location: (horizontal, depth). Location 1: (0.5, -2.0), oriented; Location 2: (-0.75, -2.5), oriented; Location 3: (-0.25, -2.0), rotated 20° to the left. Trials where the algorithm failed to navigate using segments contained within the scene have been excluded from the average computations.

Location	Success Percentage	Avg. Rotational Offset from Goal Orientation	Avg. Depth from Goal	Avg. Horizontal Displacement from Goal	Avg. Distance From Goal
Location 1 (avg. out of 9 trials)	70%	7° ± 4.74°	13.08cm ± 8.25cm	16.49cm ± 8.25cm	21.96cm ± 8.25cm
Location 2 (avg. out of 8 trials)	70%	1.38° ± 7.48°	12.21cm ± 10.09cm	-6.63cm ± 15.44cm	22.65cm ± 4.47cm
Location 3	90%	6.8° ± 4.66°	8.65cm ± 10.83cm	-4.18cm ± 7.83cm	13.48cm ± 9.44cm

The agent successfully navigated toward and stopped (approximately) at the goal in 77% of trials for the original scene. When the algorithm accurately kept track of the segment corresponding to the television box for the extent of the trial, however, the agent successfully attained the goal in 92% of trials (23/25 cases). This consistency shows that, when an agent can accurately track an object, it is using to navigate, mental rotation can effectively be used to aid in navigation. The feedforward step of the algorithm can be attributed, at least in part, to 71% (5/7 cases) of failure in the first test scene.

The feedforward step of the navigation algorithm failed to identify a matching segment contained within the scene, for at least a portion of the trial, in 10.0% of all cases during the testing of the first scene. The algorithm found no matching segment in one case and human intervention was required to stop the agent. The matching segment was not found because it was not contained within the depth range considered by the feedforward step of the algorithm. The estimation of where the segment should appear in depth was inaccurate. In the other cases, the depth at which the algorithm was looking for the box segment was again incorrect and an incorrect segment, not related to the scene, was matched with the box. The robotic agent navigated away from the goal entirely, treating this improper segment as the segment corresponding to the box.

All of the initial RGBD image captures during the testing of this scene occurred at or beyond 4m. The box was located 2m beyond the goal location, and the agent had to travel at least 2m in depth to reach the goal. This distance is at the edge of the range that the Kinect can be depended upon to accurately assess depth.

The notion that the noise from the Kinect poses an issue is supported by the 6.7% of cases where the algorithm failed to align the agent's orientation to its goal orientation due to incorrectly matching the box segment with another segment contained within the goal view. In one case, superpixels from the background and foreground merged with the "key" objects in the scene causing a misidentification to occur. In the other case, an early misidentification took the agent off course. Though the algorithm correctly identified the box in its next iteration and kept track of the box throughout the rest of the trial, the agent was unable to recover and the trial resulted in failure.

In the two cases (6.7% of all cases) where the agent did not "succeed", as defined above, and the feedforward step cannot be attributed to the failure, the agent stopped within 0.5m of the goal location. The agent failed during these two trials because it did not appropriately orient itself at the conclusion of the trial even though the algorithm had instructed it to do so. The algorithm should capture a final image to ensure that the agent has oriented itself.

4.2 More Complex Scene

The goal view of the second tested scene is shown as part of Figure 2A. This second round of testing was meant to reveal more about the capabilities of the navigation algorithm itself by placing the agent at starting positions where the Kinect would provide less noisy RGBD images. In this complex environment, there were numerous objects that were decidedly non-planar meaning that the segmentation could not be trusted to be consistent. There were planar objects at varying depths in the background of the scene. The segment the agent used to navigate, the whiteboard, was partially obscured by chair arms, and it was not at a constant depth. At the goal, the whiteboard was not entirely contained within the agent's view, while the starting location was always far enough back for the agent to be able to view the whiteboard in its entirety. The results for all three locations are summarized in Table 2, which is shown below.

The results of these thirty trials support the notion that mental rotation can be used for navigation. The agent succeeded in 90% of trials (27/30 cases) for this scene compared to 76.67% of trials (23/30 cases) in the other scene. This is likely due to the fact that the robot started closer to the scene and noise from the Kinect did not play a role. The feedforward step only failed once out of the thirty trials. In this one case, the feedforward step identified a segment composed of the floor directly below the whiteboard and the small barrel to the side of the whiteboard as the whiteboard segment. The agent failed to come within 0.5m of the goal in the second scene twice (in 6.67% of all trials). In these two trials, the agent failed to orient itself when it first began to move (though the algorithm accurately assessed the angle it needed to turn to align). It oriented itself with the next update, but the agent was unable to move quickly enough to the left or right without losing track of the segment being used to navigate to reach the goal location. Ultimately, in order to overcome the Kinect's limited field of view, different hardware will have to be used, or the algorithm will have to incorporate a memory that "remembers" the segment's position, even if it is not within view, so it can be identified when the agent needs to confirm its relative position.

Table 2. Results: Scene 2. Location: (horizontal, depth). Location 1: (0.5, -1.5), rotated 10° to the right; Location 2: (-0.75, -1.75), rotated 15° to the right; Location 3: (0.0, -2.25), oriented

Location	Success Percentage	Avg. Rotational Offset from Goal Orientation	Avg. Depth from Goal	Avg. Horizontal Displacement from Goal	Avg. Distance From Goal
Location 1	80%	-1.5° ± 6.09°	4.34cm ± 11.74cm	34.19cm ± 10.19cm	36.6cm ± 9.47cm
Location 2	90%	-1.4° ± 5.41°	16.89cm ± 14.03cm	-30.47cm ± 12.45cm	37.73cm ± 11.93cm
Location 3	100%	-0.4° ± 4.65°	13.26cm ± 10.03cm	9.7cm ± 15.48cm	21.09cm ± 12.86cm

5 Conclusions and Future Work

This paper has shown how a process inspired by the cognitive mechanism of mental rotation, a mechanism shown to be present in higher order primates, can be successfully incorporated into an autonomous robotic navigation algorithm. The algorithm introduced allowed the robotic agent to navigate in an informed way toward a goal location without doing any explicit planning. Navigation during almost all trials in which the agent was able to keep track of the segment it was using to navigate was sufficiently robust. Shortcomings in the feedforward aspect of the algorithm can likely be addressed by designing a more computationally efficient algorithm that is able to update more often and able to better smooth noisy input images. Future work also includes using an algorithm like the one described here in the context of advice giving. An agent can be informed to recognize particular objects and can be directed relative to these objects. Once the objects in the scene are recognized a procedure like the one described above can be used to successfully navigate. The process inspired by mental rotation will require rotating the entire object and using correpsondence between object features to assess orientation alignment rather than using normal vectors.

The navigation tasks presented above were simple in nature and had to be largely restricted due to the limitations of the Microsoft Kinect as a sensor. The navigation algorithm has been enhanced since these trials to allow for multi-waypoint navigation where waypoints were composed of multiple "key" segments. Though not yet rigorously tested, there have been successful demonstrations of this algorithm on the Pioneer 2-DX robotic platform. This improvement allows for testing in more complex, real-world environments; it helps to overcome the depth measurement limitations of the Microsoft Kinect.

Acknowledgments The Office of Naval Research, under grant 00014-11-1-0593, supported this research. The authors would also like to thank Professor Frank Dellaert and Natesh Srinivasan for their contributions to this project.

References

1. Thrun S., Burgard W., Fox D.: Probabilistic Robotics, MIT Press, Cambridge MA (2005).
2. Arkin R.C.: Behavior-based Robotics, MIT Press, Cambridge MA, (1998).
3. Menzel Jr, E. W., & Menzel, C. R.: Do Primates Plan Routes? Simple Detour Problems Reconsidered, (2007).
4. Menzel, C. R., Savage-Rumbaugh, E. S., & Menzel Jr, E. W.: Bonobo (Pan paniscus) spatial memory and communication in a 20-hectare forest, International Journal of Primatology, 23(3), 601–619, (2002).
5. Lourenco, S. F., & Huttenlocher, J.: Using geometry to specify location: implications for spatial coding in children and nonhuman animals, Psycholog. Res. 71(3), 252–264, (2007).
6. Aretz, A. J., & Wickens, C. D.: The mental rotation of map displays. Human Performance, 5(4), 303–328, (1992).
7. Bläsing, B., de Castro Campos, M., Schack, T., & Brugger, P.:Mental rotation of primate hands: human-likeness and thumb saliency, Experiment. brain res.,221(1), 93–105, (2012).
8. Khooshabeh, P., & Hegarty, M.: Representations of Shape during Mental Rotation, In AAAI Spring Symposium: Cognitive Shape Processing, (2010, March).
9. MacKenzie, D., Arkin, R.C., and Cameron, J.: Multiagent Mission Specification and Execution, Autonomous Robotics, 4(1), 29-57, (1997, Jan).
10. Kosslyn, S. M., Thompson, W. L., & Ganis, G. [The case for mental imagery]. Oxford University Press, (2006).
11. Hutcheson, A. T., & Wedell, D. H.: From maps to navigation: The role of cues in finding locations in a virtual environment. Memory & cognition, 40(6), 946–957, (2012).
12. Hegarty, M., & Waller, D.: A dissociation between mental rotation and perspective-taking spatial abilities, Intelligence, 32(2), 175–191, (2004).
13. Arkin, R.C.: The Role of Mental Rotations in Primate-inspired Robot Navigation, Cognitive Processing, 13(1), 83–87 (2012).
14. Stein, L. A.: Imagination and situated cognition, Journal of Experimental &Theoretical Artificial Intelligence, 6(4), 393–407, (1994).
15. Arkin, R.C., Dellaert, F., & Devassy, J.: Primate-inspired Mental Rotations: Implications for Robot Control, Proc. IEEE International Conf. on Robotic sand Biomimetics, (2012).
16. Arkin, R. C., Dellaert, F., Srinivasan, N., & Kerwin, R.: Primate-inspired vehicle navigation using optic flow and mental rotations, In SPIE Defense, Security, and Sensing (pp. 87560M-87560M), International Society for Optics and Photonics, (2013, May).
17. Erdogan, C., Paluri, M., & Dellaert, F.: Planar segmentation of RGBD images using fast linear fitting and markov chain monte carlo, In Computer and Robot Vision (CRV), 2012 Ninth Conference on (pp. 32-39). IEEE, (2012, May).
18. N. Srinivasan, F. Dellaert, A Rao-Blackwellized MCMC Algorithm for Recovering Piecewise Planar 3D model from Multiple View RGBD Images, to appear IEEE International Conference on Image Processing, October 2014.
19. PCL: Normal Estimation Using Integral Images, `http://pointclouds.org/documentation/tutorials/normal_estimation_using_integral_images.php`
20. Arkin, R. C.: Towards cosmopolitan robots: Intelligent navigation in extended man-made environments. Doctoral Dissertation. Univ. of Massachusetts, Amherst. 178–216, (1987).