Primate-inspired Autonomous Navigation Using Mental Rotation and Advice-Giving

Lakshmi Velayudhan and Ronald C. Arkin

Mobile Robot Lab, Georgia Institute of Technology

***Abstract*— The cognitive process that enables many primate species to efficiently traverse their environment has been a subject of numerous studies. Mental rotation is hypothesized to be one such process. The evolutionary causes for dominance in primates of mental rotation over its counterpart, rotational invariance, is still not conclusively understood. Advice-giving offers a possible explanation for this dominance in more evolved primate species such as humans. This project aims at exploring the relationship between advice-giving and mental rotation by designing a system that combines the two processes in order to achieve successful navigation to a goal location. Two approaches to visual advice-giving were explored namely, segment based and object based advice-giving. The results obtained upon execution of the navigation algorithm on a Pioneer 2-DX robotic platform offers evidence regarding a linkage between advice-giving and mental rotation. An overall navigational accuracy of 90.9% and 71.43% were obtained respectively for the segment-based and object-based methods. These results also indicate how the two processes can function together in order to accomplish a navigational task in the absence of any external aid, as is the case with primates.**

I. Introduction

Over the years, autonomous navigation has undergone tremendous developments with increasingly advanced techniques being devised. Nature holds examples of several efficient navigators and is a source of potential solutions to many of the remaining challenges that autonomous navigation currently faces. Primates are, by evolution, some of the most adept navigators in nature [1]. It is speculated that primates use mental rotation which is the cognitive process of rotating mental representations of objects, in order to successfully navigate their habitat. Arkin et al. in [2] and [3], discuss the architecture of a system that uses mental rotation in robot navigation using local depth maps generated at the current location of the agent as it navigates. Pettinati et al. in [4] describe the implementation of such a navigational algorithm which uses a series of mental transformations on the generated local depth maps in order to compute a rotational and translational vector that directs the agent to a goal location. The results obtained confirm the usefulness of mental rotation in robot navigation. Owing to the absence of maps or other navigational aids in this scenario, an external input or 'advice' assisting the navigation would be beneficial. By specifying a set of key objects and their relative positions with respect to the final goal, successful navigation of the robot can be attained. This algorithm does not assume any a-priori knowledge regarding the overall layout of the surroundings. Instead, it is focused on a few key elements in the environment that guides the navigation. An analogous situation would be instructing a person to *"Go down the hallway and take the first left"*. Even without awareness of the starting location and the absence of maps, the individual can successfully complete the task based on the advice given to them and their own observations of the environment. The use of these visual cues or abstract representations in navigation rather than explicit directions based on distance was observed by Hutcheson et al. in [5], indicating an underlying cognitive process supported by advice-giving that enables the goal to be achieved. In this paradigm, advice-giving may be used to explain the dominance of mental rotation higher up the primate evolutionary ladder. In this project, we attempt to explain the possible correlation between these two processes by unifying advice-giving and mental rotation into a navigational algorithm implemented on a robotic agent. The work in this paper builds upon the existing framework developed in our lab ([2], [3], [4]), in order to incorporate advice-giving.

A. Related Work

Mental rotation involves a cognitive representation of the object and Khooshabeh et al. in [6], showed that good human rotators relied mainly on the spatial configuration and utilized the visual information only in relevant tasks. This seems to indicate that mental rotation involves an analog transformation, which is further evidenced by Yohtaro et al. in [7]. They discussed the time dependency of mental rotation stating that time taken to mentally rotate the mirror image of an object is dependent on the angular disparity between the actual and mirror images. In order to overcome the cognitive difficulties associated with the analog transformation, humans tend to segment complex object into parts, individually manipulating each part. Analogous to this aspect, the algorithm used in this paper, focuses only on a key object, mentally transforming it instead of acting upon the entire scene itself.

Some species exhibit a cognitive process called rotational invariance which unlike mental rotation is time independent. In some primate species, these two processes coexist [8],[9]. It is posited that as hominids retreated from their arboreal environment, evolving an upright gait where the vertical reference plane gains importance, they exhibited a dominance of mental rotation over rotational invariance. We hypothesize that advice-giving offers a plausible explanation for the evolutionary preference of mental rotation over rotational invariance in higher primates. In this project we attempt to

understand the usefulness of advice-giving in conjunction with mental rotation in navigation.

There exist two primary spatial transformation strategies commonly employed by humans namely, spatial visualization (mental rotation) and spatial orientation (perspective taking). Though related, mental rotation and perspective taking are dissociated processes [10], [11], where the former strategy is preferred when a greater than 90 degree rotation is required to accomplish the task [10],[12],[13]. Several studies also evidence the existence of perspective taking capabilities in more evolved primate species like chimpanzees [10],[11]. This may be indicative of a co-evolutionary relationship between mental rotation and perspective taking. The role of advice-giving in these spatial transformation strategies is elucidated by Keyser et al. in [14]. The definition of a mutual knowledge base demarcates the perspectives of two individuals and advice-giving helps resolve any ambiguities associated with the key objects by "filtering out" objects that do not fit the specifications in the advice specified from one person to the other. For instance, in the algorithmic implementation described in this paper, when the external agent specifies advice regarding the key object as the "biggest red bucket", the information allows disambiguation of the object from other red objects and smaller buckets. In this respect, mental rotation may be employed, for instance, to identify the object an individual is referring to based on the advice that is specified. The navigation algorithm described in this paper thus draws from the concept of mutual knowledge and advice disambiguation. Trafton et al. in [15] implement perspective taking on a robotic platform and elucidates this form of disambiguation.

Research conducted by Rigal et al. in [13] offer some evidence that may further assert the correlation between advice-giving and mental rotation. Their research showed that children in the age group 5-11 years had difficulty discerning "left" and "right" when different from their egocentric perspective. This may be attributed to the lack of cognitive development that is necessary for performing mental rotation which helps in the differentiation. The gradual cognitive development that eventually allows them to accomplish the task may be derived from the numerous social interactions the children encounter, requiring them to perform some mental rotation either to identify objects or scenes. These interactions could be in the form of advice-giving.

Some of the components of advice-giving in a navigational scenario, as elucidated in [16], are shown below:

- A named destination – This is the final goal, for instance a goal object.
- Operations describing movement – E.g. Move straight, keep left
- Operations performed in relation to reference points – E.g. Turn left at the intersection

Some of the other characteristics of advice-giving involve sequential execution of the advice and the use of state-of-being verbs [16].

II. Object Recognition

The navigation algorithm consists of two components: a bootstrap phase and a feedforward phase. The bootstrap phase of the navigation algorithm consists of the object recognition component that enables the agent to identify the key object in the goal scene and use it as a navigational aid. A goal image to which the agent is expected to navigate is fed as input into the algorithm along with a 3D reconstruction of the key object in point cloud format (pcd). At the onset of navigation, the agent captures an image corresponding to its current location/starting point via an onboard Kinect.

The object recognition algorithm begins by extracting a set of SIFT (Scale Invariant Feature Transform) points from the object. These points are subsequently compared to the SIFT points extracted from the scene using a Euclidean distance measure in order to identify the key object in the scene. The invariance of SIFT to various image transformations like translation, scaling and rotation highlights properties similar to the neurons in the inferior temporal cortex of the brain that predominantly contributes to primates' object recognition [17]. Compared to alternatives like uniform sampling, this aspect makes SIFT an apt choice for the overall primate-inspired algorithm. Following the feature extraction, application of SHOT correspondence estimation enables the object - scene feature descriptors to be matched even in case of cluttered or occluded environments. RANSAC iterations help determine the optimum model describing the pose of the object in the scene. It first computes an initial model by randomly selecting a set of data points from the entire pool and validating its fit based on the number of inliers and outliers. Repeated iterations of the algorithm yields a final optimum model corresponding to the pose of the object. The implementation returns a 4x4 rotation matrix indicating the object pose which is subsequently used to generate the control vector used by the robot in the feedforward phase for navigation.

As stated in Section 1, the current image stored at the initial position of the agent can be related to the mutual knowledge described in [14]. The specification of a key object in the advice helps resolve any ambiguities, thus guiding the agent's navigation to the goal position.

III. Navigation Algorithm

The feedforward step as described in [4] first computes the occupancy grid from the depth images generated by the Microsoft Kinect for both the current and the goal images. A number of mental transformations are then applied to the current occupancy grid in order to match it with the occupancy grid generated from the goal image. A motion vector is derived from this correspondence and the robot moves a short distance forward as defined by the generated motion vector. It then repeats the process for the new current location until it has arrived at the goal location. Introduction of object recognition and advice-giving, in addition to shedding light on their possible correlation with mental rotation, helps eliminate a limitation of the bootstrap phase of the previous algorithm, namely the requirement of hand

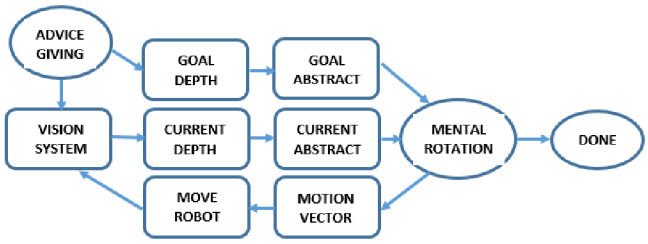


Fig. 1: Navigational algorithm architecture

matching individual segments. Fig. 1 indicates an overview of this process.

The modified algorithm can be broadly divided into two computational steps. The first involves calculation of the rotation matrix as a result of the mental transformations (mental rotation) performed by the agent on the scene captures made during the course of navigation. The second step involves the computation of a bounding box that helps the agent keep track of the object location. It also helps improve the accuracy of the object recognition algorithm by constraining the environment as described below.

A. Rotation Matrix Computation

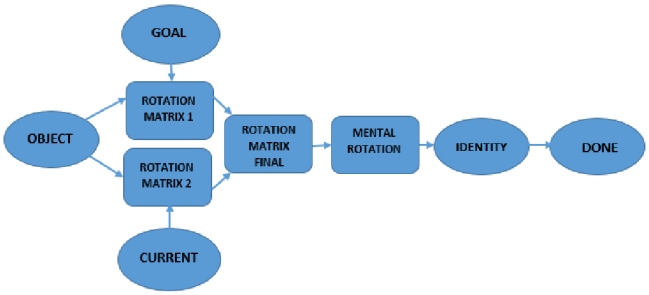


Fig. 2: Architecture of the feed forward phase

A 'relative' rotation matrix corresponding to the transformation of the object from the current to the goal scene is key to generating the motion vector for the robot's navigation and is computed from the object's pose in the current and goal images. Additionally, the resultant matrix is decomposed to isolate the rotation about the vertical axis owing to the three DOFs of the robot (X/Y translation and rotation about vertical axis).

[Obj] * [TR1] = **[Object alignment in current scene]**;
[Obj] * [TR2] = **[Object alignment in goal scene]**;
[Relative alignment] = [Tf] = (inv[TR1]) * [TR2];
Decompose [Tf] **about Y axis** = [TR-y];
Return [Tr-y];
Algorithm 1: RotCalc: Calculate final rotation matrix

Algorithm 1 (RotCalc function) summarizes this rotation matrix calculation. [TR] indicates the original rotation matrix and [Tf] the relative rotation matrix. The final rotation matrix ([TR-y]) gives the relative alignment about the Y axis, used by the remainder of the feed forward algorithm. Several iterations of this process are carried out as the agent makes a new capture at every step until it has attained the goal location. Figure 2 shows an overview of the rotation matrix computation.

B. Bounding Box Estimation

Prior to the computation of the rotation matrix, a bounding box is computed to enclose the object allowing it to be tracked at all times. Additionally, the scene surrounding the bounding box is filtered out to increase accuracy of the object recognition algorithm. The initial bounding box is currently manually drawn to include the object's expected location and is subsequently propagated automatically across successive frames depending on the rotation or translation the agent undergoes. The filtered scene is used to compute the rotation matrix. The pseudocode for the entire algorithm is shown in Algorithm 2.

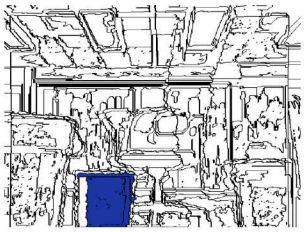
Initialize bounding box in current, goal frames;
% Compute final rotation matrix
[TR-y] = **RotCalc**([TR1,TR2]);
while *(**not** GoalReached)* **do**
  % Compute angle of rotation theta
  theta = -arcsin([TR-y](2,0));
  % Undo rotation for new bounding box location
  theta' = -theta;
  Initialize rotation matrix [TR-y]" with theta';
  Apply [TR-y]" to initial bounding box = New bounding box BB';
  % Cloud filtering step
  Extract indices outside BB';
  % New filtered cloud
  Save new cloud *Cl'*;
  if *Cl' **not** empty* **then**
    % [TR3], rotation matrix for object alignment in Cl'
    % Compute final rotation matrix with [TR3] and [TR2]
    [T-final] = **RotCalc**([TR3],[TR2]);
  else
    % Undo previous rotation
    [T-final] = [TR-y]";
  end
  Rotate according to [T-final];
  Save new current image
  [TR-y] = [T-final];
end

Algorithm 2: Feed forward pseudocode

IV. Segmentation-Based Advice-Giving

The navigation algorithm described in [4], implements segment-based navigation. The segmentation algorithm developed by Natesh et al. is described in [19]. Advice-giving was introduced into the bootstrap phase to eliminate a shortcoming of the previous algorithm, namely the requirement of hand-matching or manual cycling through all the segments in order to isolate the key segment in

(a) A sample scene with various objects



(b) Segmentation with advice specified

Fig. 3: Segmentation-based advice-giving

the goal and the matching segment in the current image. This was achieved by computing a mean RGB value and size for each segment. Two separate functions for each of these were implemented. These functions take into account all of the superpixels (groups of pixels) that compose the segment. Iterating over the pixels in each superpixel, the average RGB value (RGB data obtained from the Kinect capture) and size was computed (Total number of pixels). The advice was specified in terms of the size and color, for example, "The biggest red segment". All the segments were then filtered based on this information to finally obtain the key segment. This is based on the assumption that the segment is unique in both the current and goal images. Fig 3a shows a sample scene and Fig 3b shows its corresponding segmented image with "Biggest blue segment" specified as advice. This segment corresponds to the blue bucket.

This process is similar to the form of disambiguation illustrated in [14]. The advice specified helps eliminate any ambiguities, by filtering out the segments that do not match the description in the advice. For instance, it isolates all the blue segments and picks the largest.

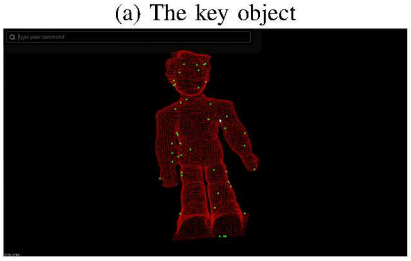
V. Results

The navigation algorithm was implemented on the Pioneer 2-DX platform. The experiments were carried out for both the segmentation-based and object-based advice-giving implementations described above. For the object case, the key object and its corresponding pcd file is shown in Fig 4.

A. Results for Segmentation-Based Advice-giving

The navigation algorithm was executed in three different test scenes shown in Figure 5. Each of them consisted of different planar and non-planar objects kept at varying depths. In each of the three scenes the final goal was the biggest red segment (the large red bucket). In location 1, there were no waypoints specified and the agent had to directly navigate to the goal segment. The goal segment was partially occluded by the blue bucket. There was a smaller red segment (small red bucket) in the background. In locations 2 and 3, one waypoint and two waypoints respectively, were specified. In location 2, the blue bucket was specified as the waypoint whereas for location 3, the first waypoint was the small red bucket and the second waypoint was the blue bucket. Fig 5 shows each of the three locations. At each of the goal positions the agent was kept facing the

(a) The key object



(b) Pcd file of the object

Fig. 4: The experimental setup

goal segment. The locations are as follows (all measurements were made from the starting position of the agent):

Location 1: The object was placed at 1.925 m and the goal position was at 1.066 m. The agent's starting position was 0.375 m to the right of the goal. No waypoints were specified.

Location 2: The goal object was placed at 3.286 m and the goal position was at 2.316 m. The agent started 0.317 m to the right of the goal. One waypoint was specified (blue segment). The blue bucket was at 1.842 m.

Location 3: The object was placed at 3.319 m and the goal position was at 3.115 m. The agent's start position was at 1.911 m to the right of the goal. The first waypoint specified was situated at 1.308 m and the second waypoint at 2.133 m.

The navigation algorithm was executed on the robotic platform until 10 successful trials were obtained. The number of failed trials that occurred before the completion of the 10 successful runs was noted as the fail rate. Hence for each location, 10 out of the total trials were successful while a varying number of failure cases were observed. Table I indicates the results obtained for the object-based advice-giving. The averages in Table I were computed with the successful trials. For the results, a '+' indicates right/ahead of goal (positive displacement) and a '-' indicates left/behind (negative displacement).

Across all the trials, the agent navigated to the goal location in 90.9% of the trials (30/33 trials). One failure case in location 2 and two failed trials in location 3 were observed. In the case of location 1, the agent made an excessive turn causing it to navigate very close to the blue segment, oriented towards the right. Since the final goal was situated to the left of this segment, eventually the red segment was lost from sight when the first waypoint was attained. In location 2 the same error was observed causing the agent to navigate very close to the first waypoint hence losing sight of the blue segment (second waypoint), requiring human intervention to

TABLE I: SEGMENT-BASED ADVICE-GIVING RESULTS

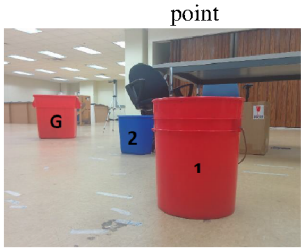
Location	**Success %**	**Avg. angular offset from goal position**	**Avg. displacement from goal**	**Avg. horizontal displacement**
Location 1 (avg. over 10/10 trials)	100 %	+ 4.04° - 4.8°	+ 14.4 cm - 7.95 cm	+ 13.86 cm - 10.3 cm
Location 2 (avg. over 10/11 trials)	90.9 %	+ 4.8° - 4°	- 10.5 cm	+ 17.3 cm - 6.8 cm
Location 3 (avg. over 10/12 trials)	83.3 %	+ 4.78° - 3.35°	+ 12.72 cm - 3.8 cm	+ 11.78 cm - 20.6 cm

TABLE II: OBJECT-BASED ADVICE-GIVING RESULTS

Location	**Success %**	**Avg. angular offset from goal position**	**Avg. displacement from goal**	**Avg. horizontal displacement**
Location 1 (avg. over 10/16 trials)	62.5 %	+ 4.33° - 2°	- 8.738 cm	+ 7.47 cm - 3.55 cm
Location 2 (avg. over 10/14 trials)	71.4 %	+ 8.37° - 1°	- 27.7 cm	+ 12.37 cm - 8.8 cm
Location 3 (avg. over 10/12 trials)	83.3 %	+ 5.75° - 5.78°	+ 18.75 cm - 23.15 cm	+ 11.53 cm - 8.63 cm

(a) Location 1 with no waypoints

(b) Location 2 with one waypoint



(c) Location 3 with two waypoints

Fig. 5: G = Goal; 1 = First Waypoint; 2 = Second Waypoint

stop the navigation.

B. Results for Object Based Advice-giving

The test scene consisted of several planar and non-planar objects kept at varying depths. The key object is the Zeno R25 Robot shown in Fig 6. The navigation algorithm was tested for three different scenarios. For each location the object was kept at a constant depth of 2.113 m. In each of these locations the object faced forward (as shown in the figure) and was displaced to the right (Location 2) and to the left (Location 3) relative to the starting position of the agent. There was no displacement for Location 1. At the goal locations the agent was oriented towards the key object. At the starting position, the agent was placed facing forward. The locations are (All measurements indicated from the starting position of the agent):

Location 1: The object was placed at 2.113 m and the goal position was at 1.463 m. The object was not displaced.

Location 2: The object was placed at 2.113 m and the goal position was at 1.489 m. The object was displaced to the right by 0.492 m.

Location 3: The object was placed at 2.113 m and the goal position was at 1.697 m. The object was displaced to the left by 0.55 m.

As previously performed, the navigation algorithm was executed until 10 successful trials were obtained. The number of failed trials that occurred before the completion of the 10 successful runs were noted as the fail rate. Table II indicates the results obtained for the object based advice-giving. The averages in Table II were computed with the successful trials. For the results, a '+' indicates right/ahead of goal (positive displacement) and a '-' indicates left/behind (negative displacement).

The agent was able to approximately navigate towards the goal position in 71.43% of the cases (30/42 cases) .The remaining 12/42 trials resulted in a failure due to the incorrect matching of the object within the bounding box

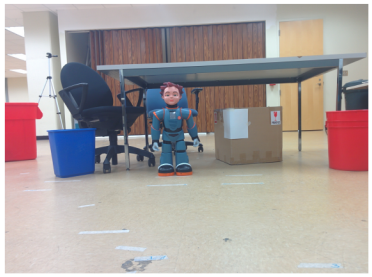


Fig. 6: Experimental scene for object-based advice-giving

which caused the agent to navigate away from the goal location. The presence of multiple objects in the scene slightly increases the chances of a mismatch. However, in situations where the object was perfectly (or almost perfectly) matched, navigation was achieved with a success rate of 100%. To some extent, as seen in the case of locations 2 and 3, higher accuracies can be achieved by correctly tuning the parameters of the object recognition algorithm to obtain better matches.

The use of bounding boxes, however, has helped improve the accuracy of the object recognition by reducing the number of mismatches and keeping the agent from straying too far from the goal position. Additionally it helped the agent keep track of the object to avoid chances of it being lost from view. For 60% of the successful trials (18/30 cases) the agent never lost the object from view. In other words, the object was present within the field of view of the Kinect. For the remaining 12 trials the agent was able to correct its position after the object was lost from view owing to a misguided rotation, using the tracking maintained by the bounding box. This led to the successful navigation of the agent towards the goal. In the cases where the navigation failed (12/42 cases), tracking was not able to correct the position of the agent since the discrepancies in the object's recognition set the agent moving much beyond the goal location. The Kinect cannot sense objects within 80 cm (2.6 Ft) depth from the sensor, rendering any correction in the position impossible once the agent enters the specified range. This causes the algorithm to run indefinitely as the subsequent captures will no longer include the key object. Hence, even though no further transformation was needed, the agent was unable to identify the actual goal position causing it to overshoot and requiring human intervention to stop the navigation.

The overall navigation algorithm is subject to significant limitations associated with the object recognition component currently employed. However, since the aim of the project is to understand the correlation between advice-giving and mental rotation by developing a system that integrates the two processes, the inaccuracies associated with the object recognition algorithm may be disregarded to some extent. As shown by the results, in situations where the object recognition algorithm has a reasonable performance (some of the incorrect matches can be compensated for using the bounding box approach), successful navigation is achieved for 100% of the cases. Even though a higher accuracy can be obtained by improving the object recognition algorithm, the existing results combined with the results obtained from the segment-based advice-giving approach certainly validate how advice-giving and mental rotation can fit together in a navigational scenario, which is the primary goal of this project.

VI. Conclusion and Future Work

This research explores a navigational algorithm inspired by mental rotation while drawing a possible relationship between advice-giving and mental rotation. It is inspired by the biological process of mental rotation by using a series of mental transformations on the observed scenes in order to achieve navigation to the goal. It incorporates the form of advice-giving elucidated in [14]. The first scene captured at the starting location of the robot is indicative of the mutual knowledge base existent between the external advising agent and the robot. Specification of a key object helps filter out the ambiguities in the advice, leading the agent to the goal location. The results obtained show that advice-giving could play a key role in guiding navigation in scenarios where maps or external aids may be absent. Additionally the system developed also indicates how advice-giving and mental rotation fit together in a navigational scenario. It should be noted that this research is intended to complement existing navigational methods such as SLAM, rather than replacing them, by providing the ability to inject advice into the navigational process.

Future work aims at trying to improve the object recognition algorithm in order to be able to correctly identify the object in a scene without the aid of bounding boxes. Additionally, waypoint specification may be incorporated into the navigation algorithm to allow the goal location to be specified in terms of a series of key objects. Future work will also focus on replacing the Kinect with a monocular camera so that its limitations regarding depth information and field of view, can be overcome and the algorithm can be tested in outdoor environments as well, which is currently limited by the capabilities of the Kinect.

VII. Acknowledgements

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

References

[1] Menzel, Charles R., E. Sue Savage-Rumbaugh, and Emil W. Menzel Jr. "Bonobo (Pan paniscus) spatial memory and communication in a 20-hectare forest." *International Journal of Primatology* 23.3 (2002): 601-619.

[2] Arkin, Ronald C. "The role of mental rotations in primate-inspired robot navigation." *Cognitive processing* 13.1 (2012): 83-87.

[3] Arkin, Ronald C., Frank Dellaert, and Joan Devassy. "Primate-inspired mental rotations: Implications for robot control." *Robotics and Biomimetics (ROBIO), 2012 IEEE International Conference* on. IEEE, 2012.

[4] Pettinati, Michael J., and Ronald C. Arkin. "A Primate-inspired Autonomous Navigation Algorithm Using the Cognitive Mechanism of Mental Rotation." *Simulation, Modeling, and Programming for Autonomous Robots.* Springer International Publishing, 2014. 388-399.

[5] Hutcheson, Adam T., and Douglas H. Wedell. "From maps to navigation: The role of cues in finding locations in a virtual environment." *Memory & cognition* 40.6 (2012): 946-957.

[6] Khooshabeh, Peter, and Mary Hegarty. "Representations of Shape during Mental Rotation." *AAAI Spring Symposium: Cognitive Shape Processing.* 2010.

[7] Takano, Yohtaro, and Matia Okubo. "Mental rotation." *Encyclopedia of cognitive science* (2003).

[8] Köhler, Ch, et al. "Mental rotation and rotational invariance in the rhesus monkey (Macaca mulatta)." *Brain, behavior and evolution* 66.3 (2005): 158-166.

[9] Burmann, B., G. Dehnhardt, and B. Mauck. "Visual information processing in the lion-tailed macaque (Macaca silenus): mental rotation or rotational invariance?." *Brain, behavior and evolution* 65.3 (2005): 168-176.

[10] Hegarty, Mary, and David Waller. "A dissociation between mental rotation and perspective-taking spatial abilities." *Intelligence* 32.2 (2004): 175-191.

[11] Kozhevnikov, Maria, et al. "Perspective taking vs. mental rotation transformations and how they predict spatial navigation performance." *Applied Cognitive Psychology* 20.3 (2006): 397-417.

[12] Zacks, Jeff, et al. "Imagined transformations of bodies: an fMRI investigation." *Neuropsychologia* 37.9 (1999): 1029-1040.

[13] Rigal, Robert. "Right-left orientation, mental rotation, and perspective-taking: When can children imagine what people see from their own viewpoint?." *Perceptual and motor skills* 83.3 (1996): 831-842.

[14] Keysar, Boaz, et al. "Taking perspective in conversation: The role of mutual knowledge in comprehension." *Psychological Science* 11.1 (2000): 32-38.

[15] Trafton, J. Gregory, et al. "Enabling effective human-robot interaction using perspective-taking in robots." *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions* on 35.4 (2005): 460-470.

[16] Psathas, George. "Direction-giving in Interaction." *Réseaux* 8.1 (1990): 183-198.

[17] Lowe, David G. "Distinctive image features from scale-invariant keypoints." *International journal of computer vision* 60.2 (2004): 91-110.

[18] Rusu, Radu Bogdan, and Steve Cousins. "3d is here: Point cloud library (pcl)." *Robotics and Automation (ICRA), 2011 IEEE International Conference* on. IEEE, 2011.

[19] Srinivasan, Natesh, and Frank 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). 2014.