Autonomous Free Flight Operations in Urban Air Mobility using Monte Carlo Tree Search

Xuxi Yang

Advisor: Peng Wei
Intelligent Aerospace Systems Lab
Iowa State University

July 19, 2018

Autonomous Free Flight Demo

Autonomous Free Flight Demo

Problem Background

- NASA, Uber and Airbus have been exploring the exciting new concept of Urban Air Mobility (UAM) [1-5], where the electric vertical takeoff and landing (eVTOL) aircraft will be autonomous for on-demand air taxi.
- The UAM operations are expected to fundamentally change cities and people's lives.
- In this paper we will combine the power of the free flight idea and onboard aircraft intelligence to enable safe and efficient flight operations.

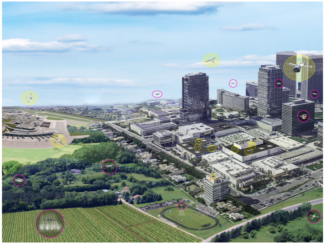


Image Courtesy: NASA UAM

Problem Background

- According to previous research on free flight [6], comparing with structured airspace, free flight with airborne separation is able to
  - handle a higher traffic density.
  - have time efficiency.
- The key to the success of the free flight is the real-time onboard computational guidance algorithm with automated conflict detection and resolution capability.

Problem Description

- Can we design a real-time computational guidance algorithm with collision avoidance capability to enable free flight enroute operations in urban air mobility?

Previous Work

- The centralized guidance or path planning algorithms generally assume complete knowledge of the world and in return provide a complete path to the destination.
- Centralized methods, which can be based on optimization technique (MILP [7], MIQP [8], SCP [9], GA [10], PSO [11], etc.) or grid-based graph search algorithm (RRT [12], A* [13]) usually generate the whole trajectory by solving one large optimization problem.
- Centralized methods can usually find the optimal solution, but it
    - can be computationally prohibitive for large fleets.
    - needs to solve the problem multiple times as new information comes.

Previous Work

- Decentralized guidance or path planning algorithms solve the conflict by each aircraft individually, based on the local information it gathered.
- Decentralized methods scale better with respect to the number of agents and are more robust against single point of failure [14].
- These methods include Potential Field algorithm [15], Deep Reinforcement Learning [16] and Geometric Approach [17].
- I will use the Monte Carlo Tree Search method to solve this problem.

Artificial Intelligence Success Stories

- Atari Games
- Robotics
- Autonomous Driving
- Knowledge and Reasoning
- Healthcare
- Dialogue Systems
- ......

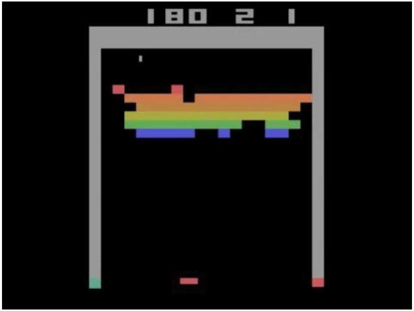


Image courtesy: Google DeepMind - DQNBreakout

Artificial Intelligence Success Stories

- Atari Games
- Robotics
- Autonomous Driving
- Knowledge and Reasoning
- Healthcare
- Dialogue Systems
- ......

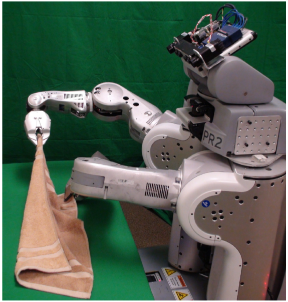


Image courtesy: UC Berkeley Robot Learning Lab

Artificial Intelligence Success Stories

- Atari Games
- Robotics
- Autonomous Driving
- Knowledge and Reasoning
- Healthcare
- Dialogue Systems
- ......

Image courtesy: Elektrobit (EB)

Artificial Intelligence Success Stories

- Atari Games
- Robotics
- Autonomous Driving
- Knowledge and Reasoning
- Healthcare
- Dialogue Systems
- ......

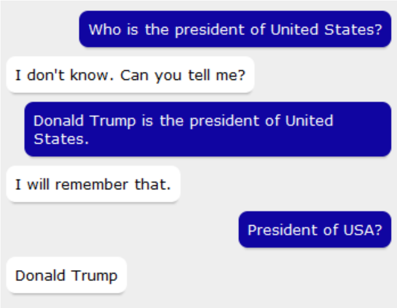


Image courtesy: Brainasoft

AlphaGo

- In March 2016, AlphaGo beat Lee Sedol in a five-game match, the first time a computer Go program has beaten a 9-dan professional.
- Monte Carlo Tree Search is the key part of the algorithm in AlphaGo.

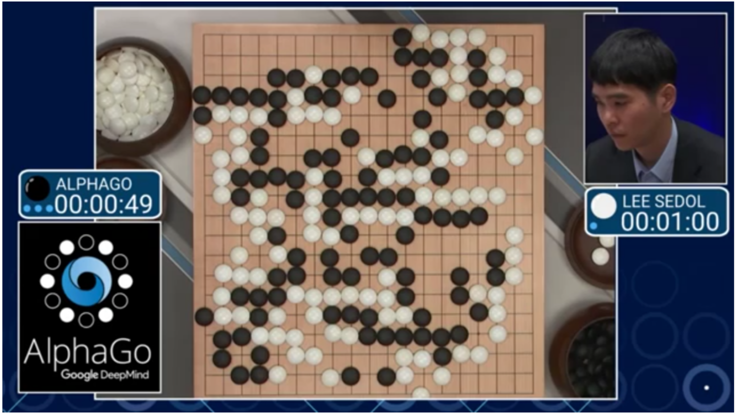


Image courtesy: Google DeepMind - AlphaGo

Our Solution Methods

- Our plan is to formulate this computational guidance problem with collision avoidance function as a Markov Decision Process (MDP)[18] and solve this MDP using Monte Carlo Tree Search (MCTS)[19].
- The goal of MDP is to maximize the reward by choosing actions optimally at each time step.
- The main idea of MCTS is to judge the reward of an action by simulations and building a search tree according to the simulation results.

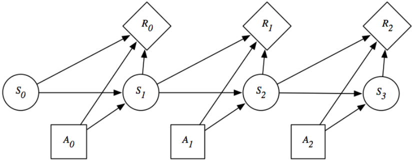


Image courtesy: [20]

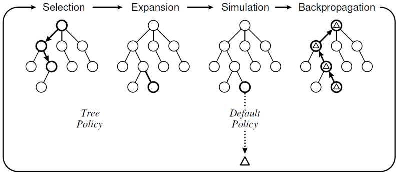


Image Courtesy: [19]

Problem Assumptions

- All the aircraft fly at the same altitude.
- All the intruders fly straight at a constant speed.
- The ownship has the perfect sensor without measurement error.

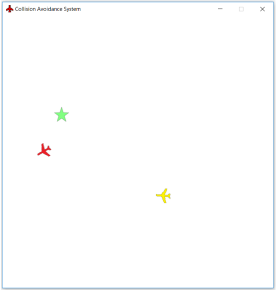
Markov Decision Process Formulation

- State: the position and velocity of all the aircraft, heading angle of ownship, and the position of the goal.
- Action: for each time step (1 second), the ownship choose to change the heading angle at a certain rate. The action space is

$$\mathcal{A} = \{-2^\circ/s, 0^\circ/s, 2^\circ/s\}$$

- Reward:
    - $+1$ if the ownship arrives at the goal state
    - 0 if there is any conflict between ownship and any intruder
- This process will terminate with reward 0 when there is a conflict.

Example of a state for MDP



- For the above figure, the state will simply be

$$s = (i_x, i_y, i_{vx}, i_{vy}, o_x, o_y, o_{vx}, o_{vy}, o_\phi, g_x, g_y)$$

Markov Decision Process Formulation

- State transition:
  - For a state

$$s = (i_x, i_y, i_{vx}, i_{vy}, o_x, o_y, o_{vx}, o_{vy}, o_\phi, g_x, g_y)$$

  - Intruder information can be decided through the assumption (they fly straight at a constant speed).
  - Goal information won't change if the ownship doesn't arrive the goal position.
  - Ownship information will be updated using kinematic based differential system.

Markov Decision Process Formulation

- State transition:
  - For a state

$$s = (i_x, i_y, i_{vx}, i_{vy}, o_x, o_y, o_{vx}, o_{vy}, o_\phi, g_x, g_y)$$

  - Intruder information can be decided through the assumption (they fly straight at a constant speed).
  - Goal information won't change if the ownship doesn't arrive the goal position.
  - Ownship information will be updated using kinematic based differential system.

Markov Decision Process Formulation

- State transition:
  - For a state

$$s = (i_x, i_y, i_{vx}, i_{vy}, o_x, o_y, o_{vx}, o_{vy}, o_\phi, g_x, g_y)$$

  - Intruder information can be decided through the assumption (they fly straight at a constant speed).
  - Goal information won't change if the ownship doesn't arrive the goal position.
  - Ownship information will be updated using kinematic based differential system.

Markov Decision Process Formulation

- State transition:
  - For a state

$$s = (i_x, i_y, i_{vx}, i_{vy}, o_x, o_y, o_{vx}, o_{vy}, o_\phi, g_x, g_y)$$

  - Intruder information can be decided through the assumption (they fly straight at a constant speed).
  - Goal information won't change if the ownship doesn't arrive the goal position.
  - Ownship information will be updated using kinematic based differential system.

Dynamic Model

- The kinematic model of the ownship is of the form:

$$\dot{x} = v\cos\phi$$
$$\dot{y} = v\sin\phi$$

- We will use the discretized version of the above model to update the ownship information:

$$
\begin{aligned}
o'_\phi &= o_\phi + a\Delta t \\
o'_{vx} &= v\cos o'_\phi \\
o'_{vy} &= v\sin o'_\phi \\
o'_x &= o_x + o'_{vx} \\
o'_y &= o_y + o'_{vy}
\end{aligned}
$$

Monte Carlo Tree Search Algorithm

- Assuming at time t, we are in state s_t, we need to iterate the following four steps to find the optimal action for current state in real time.
- **Selection step**: when we are at a state we have seen, select the action j to maximize

$$\bar{X}_j + 2\sqrt{\frac{\ln n}{n_j}}$$

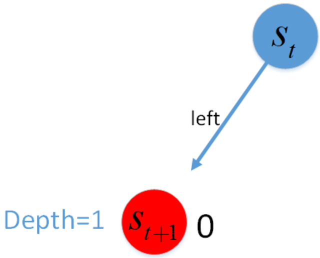
where $\bar{X}_j$ is the mean action value for action j, and n is the number of times the current state has been visited, n_j is the number of times that action j has been used. In this way we can balance exploitation and exploration.

Monte Carlo Tree Search Algorithm

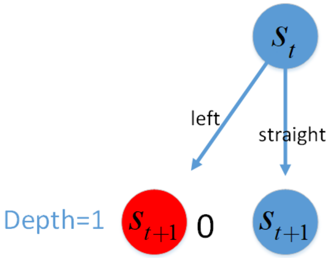
- **Expansion step**: when we meet a new state, one random action will be added to expand the tree.
- **Simulation step**: a simulation is run from the new state to a terminal state according to random policy, and then produce a reward for this terminal state.
- **Backpropagation step**: the simulation result is "backed up" through the selected states to update their value.

Monte Carlo Tree Search Algorithm

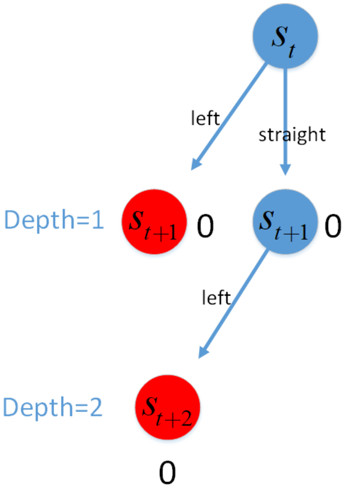
Monte Carlo Tree Search Algorithm



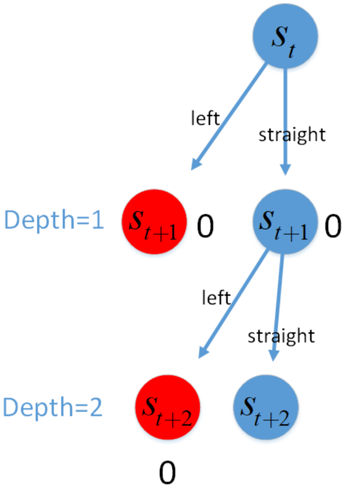
Monte Carlo Tree Search Algorithm



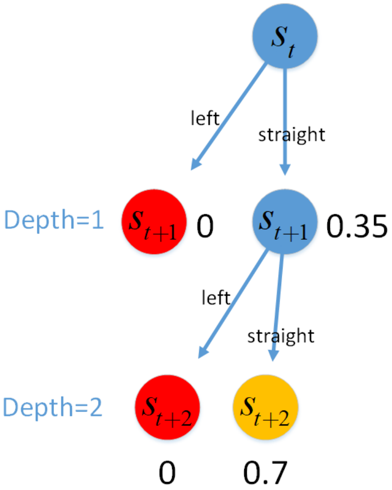
Monte Carlo Tree Search Algorithm



Monte Carlo Tree Search Algorithm

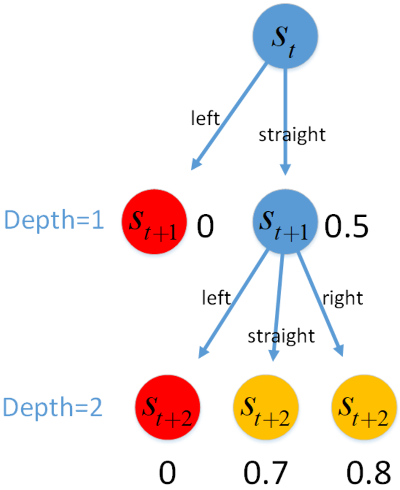


Monte Carlo Tree Search Algorithm



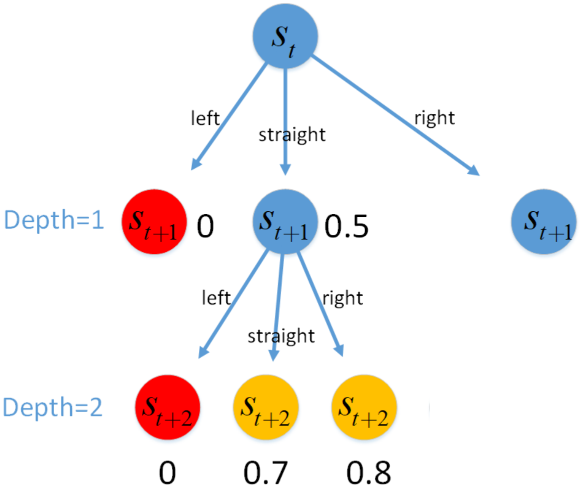
We use $1 - \frac{d(o,g)}{\max d(o,g)}$ to represent the value of these non-terminal states (depth=2).

Monte Carlo Tree Search Algorithm



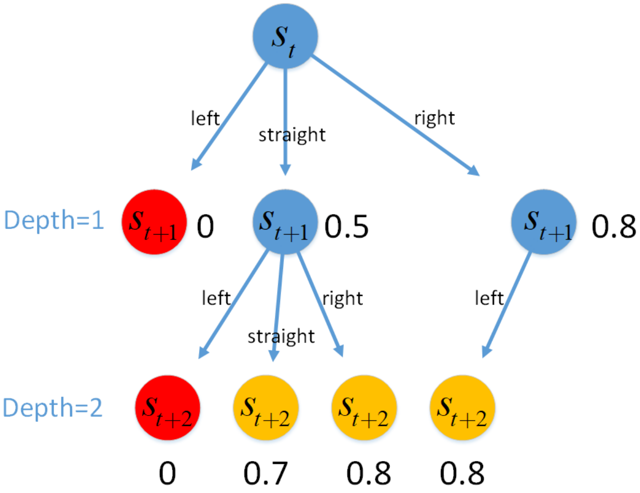
We use $1 - \frac{d(o,g)}{\max d(o,g)}$ to represent the value of these non-terminal states (depth=2).

Monte Carlo Tree Search Algorithm



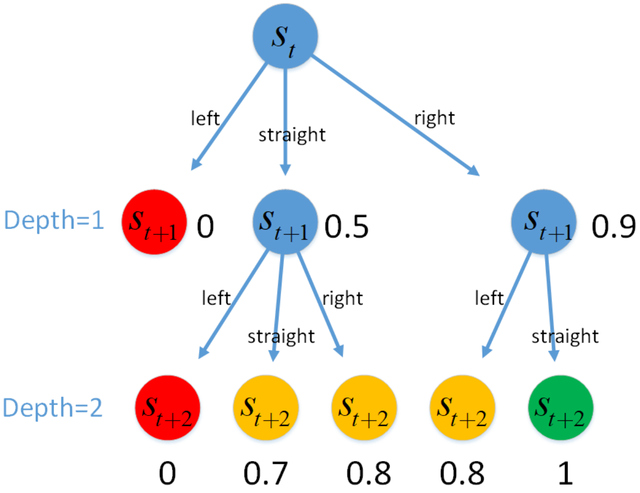
We use $1 - \frac{d(o,g)}{\max d(o,g)}$ to represent the value of these non-terminal states (depth=2).

Monte Carlo Tree Search Algorithm



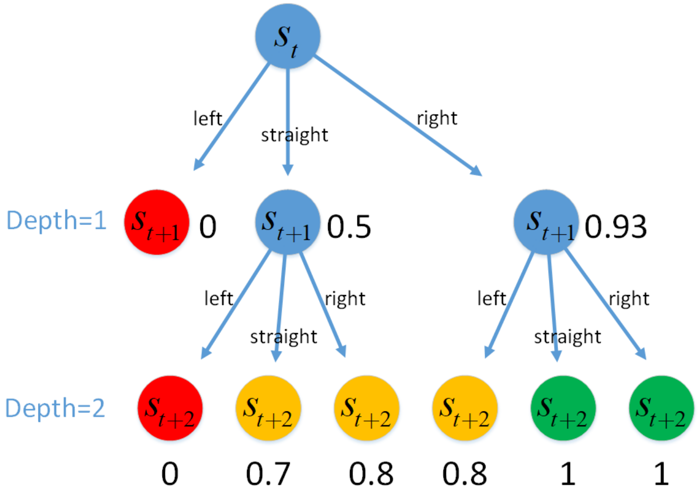
We use $1 - \frac{d(o,g)}{\max d(o,g)}$ to represent the value of these non-terminal states (depth=2).

Monte Carlo Tree Search Algorithm



We use $1 - \frac{d(o,g)}{\max d(o,g)}$ to represent the value of these non-terminal states (depth=2).

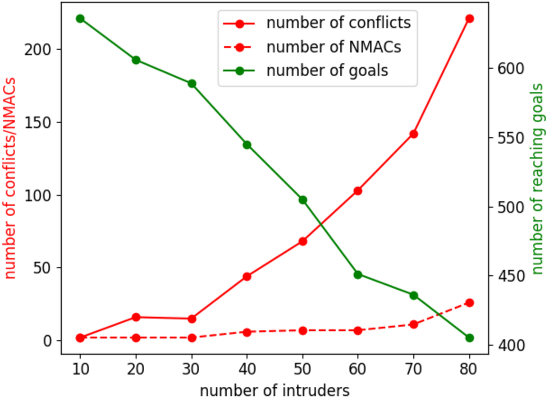
Monte Carlo Tree Search Algorithm



We use $1 - \frac{d(o,g)}{\max d(o,g)}$ to represent the value of these non-terminal states (depth=2).

Experiment Setup

Result



Conflict: when two aircraft are closer than 320m.
NMAC: when two aircraft are closer than 30m.

Conclusion

- We proposed a computational guidance algorithm with collision avoidance capability for autonomous on-demand free flight operations in urban air mobility.
- We formulate this problem as a Markov Decision Process (MDP) and solve it using Monte Carlo Tree Search (MCTS) algorithm.
- Simulation results show this algorithm has promising performance.
- The contribution of this research is integrating the power of onboard aircraft intelligence (vehicle autonomy technology) and the advantage of the free flight concept for airspace operations to enable safe and efficient flight operations in on-demand urban air transportation.

Future Work

- Allow speed change for this MCTS algorithm.
- Try to use this MCTS framework to control multiple aircraft simultaneously.
- Allow higher fidelity dynamics of the aircraft.
- Incorporate uncertainties in aircraft dynamics and the environment.

Autonomous Free Flight Demo (with Speed Change)

Autonomous Free Flight Demo (Multi-Agent)

References

1 Gipson, Lillian. 2017. "NASA Embraces Urban Air Mobility, Calls for Market Study." NASA. November 7. Accessed January 19. https://www.nasa.gov/aero/nasa-embraces-urban-air-mobility.

2 "Uber Elevate — The Future Of Urban Air Transport." 2018. Accessed January 19. https://www.uber.com/info/elevate/

3 Holden, J., and N. Goel. "Fast-Forwarding to a Future of On-Demand Urban Air Transportation." San Francisco, CA (2016).

4 "Urban air mobility." 2017. Airbus. June 20. http://airbus-xo.com/urbanair-mobility/.

5 "Future of urban mobility." 2018. Airbus. Accessed January 19. http://www.airbus.com/newsroom/news/en/2016/12/My-Kind-Of-Flyover.html.

References

6 Hoekstra, Jacco M., Ronald NHW van Gent, and Rob CJ Ruigrok. "Designing for safety: the 'free flight' air traffic management concept." Reliability Engineering & System Safety 75, no. 2 (2002): 215-232.

7 Schouwenaars, Tom, Bart De Moor, Eric Feron, and Jonathan How. "Mixed integer programming for multi-vehicle path planning." In Control Conference (ECC), 2001 European, pp. 2603-2608. IEEE, 2001.

8 Mellinger, Daniel, Alex Kushleyev, and Vijay Kumar. "Mixed-integer quadratic program trajectory generation for heterogeneous quadrotor teams." In Robotics and Automation (ICRA), 2012 IEEE International Conference on, pp. 477-483. IEEE, 2012.

9 Augugliaro, Federico, Angela P. Schoellig, and Raffaello D'Andrea. "Generation of collision-free trajectories for a quadrocopter fleet: A sequential convex programming approach." In Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on, pp. 1917-1922. IEEE, 2012.

References

10 Delahaye, Daniel, Clément Peyronne, Marcel Mongeau, and Stéphane Puechmorel. "Aircraft conflict resolution by genetic algorithm and B-spline approximation." In EIWAC 2010, 2nd ENRI International Workshop on ATM/CNS, pp. pp-71. 2010.

11 Pontani, Mauro, and Bruce A. Conway. "Particle swarm optimization applied to space trajectories." Journal of Guidance, Control, and Dynamics 33, no. 5 (2010): 1429-1441.

12 Kothari, Mangal, Ian Postlethwaite, and Da-Wei Gu. "Multi-UAV path planning in obstacle rich environments using rapidly-exploring random trees." In Decision and Control, 2009 held jointly with the 2009 28th Chinese Control Conference. CDC/CCC 2009. Proceedings of the 48th IEEE Conference on, pp. 3069-3074. IEEE, 2009.

13 Xia, Li, Xie Jun, Cai Manyi, Xie Ming, and Wang Zhike. "Path planning for UAV based on improved heuristic A* algorithm." In Electronic Measurement & Instruments, 2009. ICEMI'09. 9th International Conference on, pp. 3-488. IEEE, 2009.

References

14 Pallottino, Lucia, Vincenzo Giovanni Scordio, Emilio Frazzoli, and Antonio Bicchi. “Probabilistic verification of a decentralized policy for conflict resolution in multi-agent systems.” In Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on, pp. 2448-2453. IEEE, 2006.

15 Sigurd, Karin, and Jonathan How. “UAV trajectory design using total field collision avoidance.” In AIAA Guidance, Navigation, and Control Conference and Exhibit, p. 5728. 2003.

16 Chen, Yu Fan, Miao Liu, Michael Everett, and Jonathan P. How. “Decentralized non-communicating multiagent collision avoidance with deep reinforcement learning.” In Robotics and Automation (ICRA), 2017 IEEE International Conference on, pp. 285-292. IEEE, 2017.

17 Van Den Berg, Jur, Stephen J. Guy, Ming Lin, and Dinesh Manocha. “Reciprocal n-body collision avoidance.” In Robotics research, pp. 3-19. Springer, Berlin, Heidelberg, 2011.

References

18 Bellman, Richard. "A Markovian decision process." Journal of Mathematics and Mechanics (1957): 679-684.

19 Browne, Cameron B., Edward Powley, Daniel Whitehouse, Simon M. Lucas, Peter I. Cowling, Philipp Rohlfshagen, Stephen Tavener, Diego Perez, Spyridon Samothrakis, and Simon Colton. "A survey of monte carlo tree search methods." IEEE Transactions on Computational Intelligence and AI in games 4, no. 1 (2012): 1-43.

20 Poole, David L., and Alan K. Mackworth. Artificial Intelligence: foundations of computational agents. Cambridge University Press, 2010.