The Coordination of Deliberative Reasoning in a Mobile Robot

Patrick Ulam, *Student Member, IEEE*

Abstract— **This paper examines the problem of how a mobile robot may coordinate among multiple, possibly conflicting deliberative processes for reasoning about object interactions in a soccer domain. This paper frames deliberative coordination as an instance of the algorithm selection problem and describes a novel framework by which a mobile robot may learn to coordinate its deliberative reasoning in response to constraints upon processing as well as the performance of each deliberative reasoner. Results of the framework are described for a simulated soccer task in which the robot must predict the motion of a fast moving ball in order to prevent it from reaching the goal area.**

I. Introduction

A hallmark of human deliberative activity is its flexibility, in part afforded by the multiple, overlapping forms of reasoning found in domains ranging from spatial and physical reasoning to logical reasoning. For example, four independent ways of reasoning about object location or three independent forms of physical reasoning may be applied at different times and under different circumstances (e.g. [1]). In almost all cases, different forms of reasoning within a particular domain differ in their accuracy, knowledge representation, or cognitive resource consumption. These differences allow humans to adapt their high-level reasoning in a manner that is in alignment with their current goals, cognitive resources, and environmental constraints.

Being embedded in a physical environment, mobile robots must plan and reason under similar constraints. Their processing is inherently limited by their hardware configuration and interaction with the physical world may place hard temporal constraints upon both action and deliberative activity. Because of this, it is likely that mobile robots would also benefit from leveraging suites of deliberative reasoners with differing performance characteristics. Solely equipping a mobile robot with multiple deliberative processes will not necessarily provide significant benefit, however. It is necessary to identify among these capabilities which deliberative process is appropriate given the robot's current goals, abilities, and the constraints under which deliberative activity will occur. We term this deliberative coordination.

Providing mobile robots with a means of coordinating between multiple deliberative processes would result in a number of benefits. The most important of these benefits being an increase in the robot's ability to pursue and achieve high-level goals in a wider variety of environments. With a framework capable of identifying the need for, the appropriateness of, and the effectiveness of particular deliberative processes, the robot can increase its flexibility in pursuing high-level goals by adapting its reasoning to meet the temporal or accuracy constraints inherent in embodied, real-time, robotic systems. Further, a principled framework for the use of multiple deliberative processes in a mobile robot will increase the robustness of the robot by better preparing it to react to failures in its planning or reasoning.

This paper presents a framework by which a mobile robot may learn to coordinate between multiple deliberative processes, each of which may differ in computational complexity, accuracy, and knowledge requirements. The framework is described and evaluated within the context of a simulated robotic soccer domain. Within this domain, the benefit of the deliberative coordination framework in terms of task performance and resource minimization is demonstrated.

II. Related Work

Within the artificial intelligence community, it has been long recognized that for many reasoning problems, solution quality and the time spent obtaining a solution can vary greatly based on the algorithm, the representation of the problem, the algorithm parameters, and numerous other factors [2]. Similar results are described within machine learning through the 'No Free Lunch' theorems developed for both supervised learning and search [3]. Evidence such as this suggests that leveraging suites of reasoning or learning algorithms could serve to improve system performance by shoring up the weaknesses of one particular algorithm with the strengths of another.

In order to leverage suites of deliberative reasoners, it is necessary to identify which planner is appropriate for the current goals of the robot. While this problem has not been significantly explored in terms of robotic systems, a growing body of work in the planning, machine-learning, and meta-reasoning communities has been investigating coordination from the perspective of algorithm selection [1][4-6]. Algorithm selection, a necessary component of many forms of deliberative coordination can be stated as: given a problem set, a set of algorithms capable of solving elements of the problem set, and a performance metric, what is the mapping between the problem set and algorithms that maximizes the performance metric? As this problem is undecidable in its most general terms [5], a number of researchers have examined how systems can learn an approximate mapping from experience.

Author is with the Mobile Robot Laboratory of the College of Computing at the Georgia Institute of Technology, Atlanta, GA, USA. (e-mail: pulam@cc.gatech.edu).

One such system, created by Fink, selected between multiple search strategies in the Prodigy planning framework [7]. Fink used a reward-based heuristic in which a particular search strategy would receive a reward for solving the search problem, discounted by the time spent solving it. These rewards were then used to guide selection for later problems. Within that work, however, only the performance statistics over the entire problem space were measured. Thus in this work, mappings from the entire problem space to a single deliberative process were made.

Other work in algorithm selection has attempted to learn a more fine-grained mapping between individual problems and deliberative planners. The approach typically taken and that used in this work is to use sets of problem features to delineate groups of problems from one another. A mapping between these problem features and deliberative planners are then learned based on statistical performance metrics. Typically the problem features used tend to be domain general features describing the structure of the input problem (e.g., the number of predicates in the input or the –arity of the predicates).

Examples of this approach include Leyton-Brown who examined the use of linear-regression to identify the relationship between domain specific problem features and algorithm run-time in order to select the algorithm best suited for a particular problem instance [8] and Gagliolo and Shmidhuber examined algorithm selection in the context of an n-armed bandit problem [2].

Most of these existing approaches to algorithm selection typically utilize isomorphic problem, solution, and knowledge representations. Because of this, in much of this existing work, the primary difference between the available algorithms/planners lies in the time necessary to generate a solution. In robotics systems, may be desirable to not only select deliberative process based upon the time necessary to generate a solution but also upon the fidelity of the solution necessary for a particular task.

III. A Framework for Deliberative Coordination in a Mobile Robot

This section outlines the framework used for deliberative coordination in this work and how it is integrated within a three-tier mobile robot architecture. The deliberative coordination process presented can be roughly divided into three phases: problem clustering, process selection, and process evaluation. Problem clustering is the process of grouping the set of available input problems addressable to the robot's deliberative reasoners into a finite number of problem classes. A problem class is defined as a collection of problem features, where problem features are finite valued elements which serve to characterize the current input to the deliberative processes. Each unique valuation of problem features corresponds to a different problem class. Figure 1 shows the relationship between problem features and problem classes. Problem features can be any set of sensory precepts or abstract meta-features that describe the current goal of the robot. For example, if the robot's goal was to transport an object to another location, problems features may consist of percepts describing the environment or meta-features describing the actual input to the planner that would generate the necessary plan (e.g. the number of predicates in the planner input). Whatever the problem features may be, however, they must potentially serve as predictors of the planners' or other deliberative processes performance on the input.

Each problem class has associated with it a set of process profiles (more commonly known as algorithm profiles in the any-time algorithm literature). These process profiles store information about the estimated quality of solution that will be generated by one of the robot's deliberative process as a function of the time the robot spends reasoning with that process (e.g. Figure 2). These may be given by the designer or learned from experience. One technique by which the robot may learn these profiles is described later in this section. These process profiles can be formalized as the function $Q(C,d,t)$. This function can be interpreted as the estimated solution quality for problems of class *C*, as addressed by deliberative process *d* for *t* time units, where the output of *Q* is a real number. Once the appropriate set of profiles has been indexed by the problem class, the robot may use this collection of profiles to select the deliberative reasoner appropriate for its goals and any constraints upon reasoning.

The final step in coordination is process selection. During process selection, the robot activates one of the deliberative processes to reason about the robot's current goal based upon their performance profiles (*Q)* for the currently identified problem class and any constraints upon reasoning. The constraints are used to determine the function used to select the appropriate deliberative process. For example, if there no were temporal constraints upon the solution time the following function would be used to select the deliberative process estimated to produce the highest quality solution: $\arg\max_x Q(C,d_x,\infty)$ (d_2, Figure 2). Selecting the deliberative process estimated to produce the highest quality solution by time *t*, on the other hand would be formulated as $\arg\max_x Q(C,d_x,t)$ (d_1, Figure 2), while

Problem Feature	Values
f_1	*a, b*
f_2	*c, d*

Problem Class	Feature Values
c_1	(*a,c*)
c_2	(*a,d*)
c_3	(*b,c*)
c_4	(*b,d*)

Figure 1. Depicts the relationship between problem features and problem classes. A problem class is defined by a unique collection of problem features.

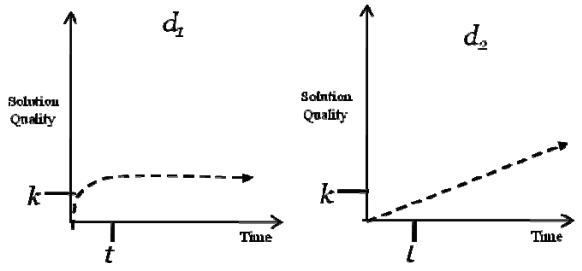

Figure 2. Sample process profiles used to perform deliberative coordination.

selecting the fastest deliberative process estimated to produce a solution of quality k would be formulated as: $\arg\max_x [\arg\max_t (Q(C, d_x, t) \geq k)]$ (d_1, Figure 2).

In most cases, the quality of solution that would be obtained using a process on a problem class would not be known to the robot *a priora*. If these values are not known by the designer, the robot may be able to learn these values from experience. In order to do this, each problem class and duration tuple are treated as single state reinforcement learner (n-armed bandit) in which each of the actions available from that state correspond to the selection of a deliberative process for that problem class. After a deliberative process has been selected for a particular problem, the robot may evaluate the output of that process and the evaluation will serve as feedback to learn the deliberative process profiles. The feedback will depend on the exact output of the deliberative process but will pertain to the measured quality of solution generated by the process. If the process output were a plan to be executed by the robot, the evaluation may be based on the success of the plan or how quickly the plan is accomplished. If the process were to output a prediction about the future state of the world, the evaluation may be based on the accuracy of the prediction.

This feedback signal can then be used to update the stored process profiles used by the robot for coordination. In this work, this update occurs by computing the exponential, recency-weighted average of the feedback signals. This may be formulated as follows: $Q_{k+1}(C, d_i, t) = Q_k(C, d_i, t) + \alpha[r - Q_k(C, d_i, t)]$, where k, is the time, C, is the problem class, d is the selected deliberative process, and t is the amount of processing time allocated for reasoning by the robot.

Figure 3 depicts how the deliberative coordination framework described may be incorporated into a typical three-layer hybrid reactive-deliberative architecture (see [9] for a review). The deliberative layer contains one or more deliberative processes that are used by the robot to generate plans for the system or otherwise reason about the robot's goals. The deliberative processes are activated via the coordination subsystem implemented as described earlier in this section. Within the coordination module are the process profiles describing the performance

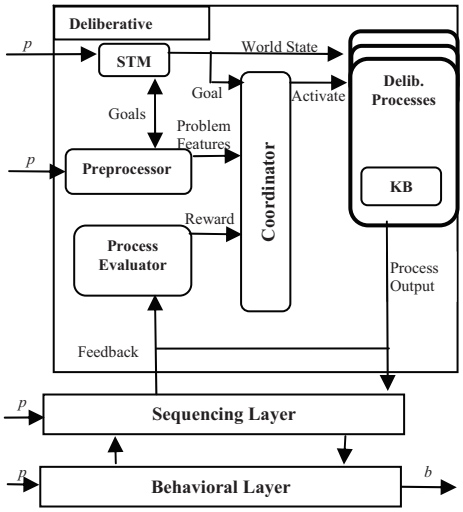

Figure 4. High level architectural overview of coordination framework.

characteristics of the robot's available deliberative processes. The preprocessor module transforms incoming perceptual data into problem features which the coordinator may use to select the correct collection of process profiles that will be used for deliberative coordination. Any plans generated by the deliberative layer are sent to the sequencing layer for realization by the underlying behavioral control system of the robot. The sequencing layer also monitors the progress of any plans generated by the deliberative system in order to provide feedback to the evaluation module. Finally, this feedback is transformed into a reward signal which the coordinator may use to update the process profiles stored therein.

IV. Simulation Experimental Verification

In order to evaluate the deliberative coordination framework, a series of simulation experiments were conducted within a robotic soccer domain. The simulation environment used was custom built and incorporated NVidia's PhysX dynamics engine to ensure realistic physical behavior from both the robot and environment. In this domain, a single goalie robot must prevent a fast moving ball from entering a goal area behind the robot. The environment in which this task takes place is a 4m by 5m enclosed area. In addition to the goalie robot, within this environment, 10 *static* players are placed throughout. These players do not actively interact with the ball but only serve as objects which may alter the motion of the ball upon collision (Figure 4).

During this task, at 200ms intervals, the robot would select one of its deliberative processes to predict the future trajectory of the ball as well as the amount of time that process would spend computing this prediction. The selected process would then compute the ball's trajectory, and if the process predicts the ball would enter the goal area, a plan would be generated for intercepting the ball. At the end of this 200ms cycle, the process's prediction is then evaluated based on the actual distance of the ball

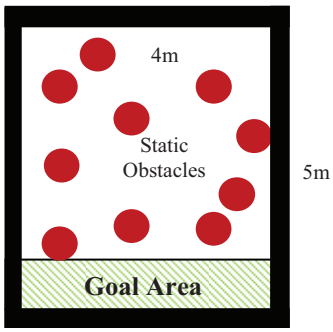

Figure 5. Typical environment used in experimental evaluation. Red circles represent static players/obstacles in the field.

from the predicted trajectory. This error is fed back to the coordination module as described in Section III. The three processes used for ball prediction will be detailed in the following section.

The robot that serves as the goalie is based on the specifications of a typical small-league soccer robot 20cm in diameter with a maximum velocity of 2m/s and maximum acceleration of $4m/s^2$. The ball that must be prevented from entering the goal area is highly elastic (coefficient of restitution 0.9), and begins each trial moving at a speed ranging from 6 m/s to 8 m/s at a random heading and position 4m from the goal area.

A. Deliberative Processes

In the simulation experiments conducted, the robot was equipped with three different processes to predict the trajectory of the ball. Each of these processes differ in the fidelity of prediction, the internal knowledge representation used to predict the future trajectory of the ball, and the amount of processing necessary to compute this future trajectory. No claims as to the appropriateness of these processes or their knowledge representations are made by this work. The processes used have been chosen solely so that they will differ from one another in terms of their computational cost when used by the robot and the accuracy of their predictions. This is been done in an effort to simplify the evaluation of the presented framework and other processes may be substituted without loss of generality.

The three processes used are termed the binary, qualitative, and quantitative processes respectively. Their names reflect the manner in which each process reasons about the motion of the ball and its reactions to collisions within the environment. Each process is responsible for predicting the future trajectory of the ball and if that predicted trajectory results in the ball traveling to the goal area, generating a plan for the robot to intercept the ball. Details as to how each process do this follow.

The binary process is the simplest of the three. It makes its predictions about the future path of the ball based solely upon one factor, if the ball's current path will enter the goal or not. This simple reasoner assumes the ball will continue moving along its current path until it reaches an object in the environment (wall or static player) or the goal. If the ball will collide with the environment, no further prediction can be made by this process, so no interception plan is generated. If the ball's current path will take it to the goal area, a plan is generated to move the robot to an interception point just outside of the goal area.

The remaining two process utilize information about the current motion of the ball and the direction of any potential collisions to predict the future ball trajectory. To do this, the qualitative and quantitative process both use the current motion of the ball (in terms of its horizontal and vertical velocity) and the direction in which the ball may collide with the environment (represented as the collision contact normal). These two elements are shown in Figure 6.

These processes then use this information to search their knowledge base for information pertaining to collisions with similar characteristics. Once knowledge considering a similar collision is found, it is used to extrapolate the motion of the ball after the collision. Both the qualitative and quantitative process may then use the resulting prediction recursively in order to increase the resulting prediction horizon (e.g. trajectory after 2 or more collisions).

The primary difference between the qualitative and quantitative process lies in the internal format used to represent the ball and collision information as well as the knowledge used to describe ball-environment interactions. The qualitative process represents ball motion, collision normal, and knowledge using a sign calculus [10]. The quantitative process, however, stores these values in absolute numeric terms. Examples of the internal representations used by the qualitative and quantitative process are shown in Figure 7.

The time necessary for each deliberative process to compute its predictions is based upon the amount of knowledge each process must search in order to find a relevant match. The qualitative process must execute approximately 100 search operations in order to find a matching knowledge element. The quantitative process must search approximately 300 knowledge elements. For the purposes of this experiment, one knowledge element may be examined in 0.1ms. In addition, each process is assumed to result in 1ms of unavoidable overhead. As a result, time necessary to predict a trajectory involving a chain of one, two, or three ball-environment interactions is shown in Figure 8.

B. Simulation Experiment I

Two experiments were conducted. The first experiment examined the ability of the system to learn

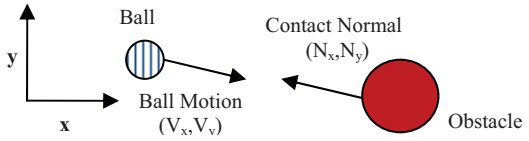

Figure 6. The qualitative and quantitative deliberative processes use qualitative and quantitative representations of the ball velocity and contact normals in order to predict the ball's motion and generate an interception plan.

estimates of deliberative process solution quality (process profiles) based on a limited number of perceptual features describing the current instance of the ball interception task. The features used in this and experiment two are shown in Figure 9. In order to learn the process quality profiles, the robot executed 180 trials of the interception task in which it used *one* of its deliberative processes to predict the ball trajectory and generate an appropriate interception plan. A prediction and plan was made every 200ms. After each 200ms cycle, the prediction generated by the process was evaluated by measuring the Euclidean distance between the predicted position of the ball and the actual resulting position of the ball. This error in prediction then served as a feedback signal to the coordination framework as described in Section III. After each set of 30 trials, the process profiles learned by the robot (Q) were compared against this evaluation metric (the distance between the ball's location predicted by a particular deliberative process and its actual position). This serves as a measure of how accurate the robot's learned process profiles are to the actual solution quality generated by each process in a particular 200ms interval of the ball interception task. This training period was then repeated for each of the three deliberative processes.

C. Simulation Experiment II

The second experiment examines the utility of the proposed coordination framework in terms of task performance and resource minimization. Using the process profiles learned in experiment one, the robot executed 20 runs of the ball interception task. Each run consisted of 100 trials. During the execution of each trial in experiment two, however, the robot would use the learned process profiles to select which deliberative process to use for predicting the ball trajectory and generating the appropriate interception plan. The deliberative process for each 200ms interval was selected using the process profiles so that the highest quality prediction was made in the least amount of time. This process is discussed in Section III. After the process to generate the interception plan is selected, the robot follows the plan until the start of the next 200ms cycle in which the coordination process begins once more. The trial ends when the ball is intercepted, enters the goal area, or stops moving. The robot's performance in the interception task is measured in terms of the number of successful interceptions and the total amount of time spent generating plans to intercept the ball. The robot's performance using the proposed coordination framework was compared against three controls: a robot that always used the fastest deliberative process (binary process), a robot that always used the deliberative process with the most accurate predictions (quantitative process for 10ms), and a robot that randomly selects the process to generate the interception plan.

Representation	Ball Motion (Vx, Vy)	Contact Normal (Vx, Vy)	Resulting Motion (Vx, Vy)
Qualitative	(+,-)	(-.+)	(-.-)
Quantitative	(2.0,-10.0)	(-0.71, 0.71)	(-7.7, -2.1)

Figure 7. Example of the knowledge representation used by the qualitative and quantitative deliberative processes.

Deliberative Process	1 Collision	2 Collision	3 Collision
Binary	1ms	N/A	N/A
Qualitative	2ms	3ms	4ms
Quantitative	4ms	7ms	10ms

Figure 8. The time necessary for each deliberative process to predict the ball's motion and generate an interception plan for a trajectory that results in the specified number of collisions.

Problem Feature	Description	Values
Ball Distance	Distance between the ball and the robot	**Short** (0 to 1m) **Medium** (*1 to 2m*) **Far** (*>2m*)
Ball Speed	Speed of the ball within the environment	**Slow** **Medium** **Fast**
Impeding Wall Collision	Will the ball's current trajectory result in a collision with the wall	**True** **False**
Impending Object Collision	Will the ball's current trajectory result in a collision with an object	**True** **False**
Direction of Ball Movement	Is the ball moving towards the goal or not.	**Towards Goal** **Away from Goal**

Figure 9. Problem features and used in the ball interception task.

V. Results

For experiment 1, the root mean squared error between the learned process profiles and the actual solution quality resulting from using a deliberative process in experiment one is shown in Figure 10. As can be seen, the estimated solution quality for each of the possible process profiles converges to between one and two. This indicates that the robot successfully learns the quality of prediction to expect from each deliberative process. While the process quality estimates learned by the system converge to values reasonably close to the actual values, and these estimates improve with experience, they alone are not suitable to determine if the process profiles learned are of any value to the robot. In order to determine this, the performance of the robot when using these estimates for deliberative coordination needs to be examined. The results from experiment two shown in Figures 10 and 11 provide such an examination.

Depicted in Figure 11 are the statistics concerning the performance of the robot in the object interception task measured in terms of the percentage of successful ball interceptions that occurred. The bars in this figure denote 95% confidence intervals. As can be seen in the figure, the proposed coordination framework results in the highest success rate for the object interception task (70%). It should be noted, however, that the difference between the task success rate for the proposed framework and control which always selected the most accurate process (the quantitative process applied for 10ms) is not statistically significant. It should also be mentioned that in a number of trials (~20%), interception failure were due to the physical limitations of the platform. In these

trials, when the ball was initialized at a high velocity (7-8m/s) and with a unobstructed initial heading directed towards the goal, the physical characteristics of the robot were the limiting factor preventing interception and not necessarily the deliberative processes (e.g. the robot was physically unable to travel to the interception location in time). These high ball velocities were deemed acceptable, however, in order to ensure that the task remained both reasonably challenging and time critical.

While the performance of the coordination framework and the accuracy-based control were not statistically significant, the computational cost of the two (measured in total computation time), significantly differ. The computational load of each system is shown in Figure 12. As can be seen in this figure, the total time spent by the robot performing reasoning for each set of 100 trials when using the coordination framework is almost half that used by the accuracy- based control. This result, coupled with those shown in the previous figure demonstrate that the deliberative coordination in a mobile robot may have a significant impact both in terms of task success and in resource minimization.

VI. Discussion and Conclusion

The simulation results depict that the proposed coordination framework results in performance matching that of the most accurate control in the domain tested while reducing the amount of computation necessary to achieve that performance. The benefit afforded by the coordination framework, however, is dependent on the properties of the domain in which the robot will be deployed. In the domain examined, robot soccer, high performance is preferable. Thus, deliberative behavior that maximizes performance while minimizing the computation necessary to achieve that performance will allow the robot to spend additional processing time upon other tasks (e.g. perception). Alternate domains, however, may place a premium on computation reduction (and its potential power savings). Such tradeoffs are domain dependent but can be incorporated into the proposed framework by modifying the process selection function described in Section III.

While flexible in this regard, the coordination framework also has several limitations. In particular, coordination requires the robot possess knowledge concerning the performance characteristics of its available deliberative processes. This knowledge may be difficult for the designers to provide and thus necessitate significant amounts of training. As a result, the approach is likely best suited to domains in which such training is feasible. Exploring the sensitivity of the framework to inaccurate or incomplete performance estimates may help better characterize this limitation and is one of several avenues of future research that are being pursued.

References

[1] Newcombe, N. S. and J. Huttenlocher , Making Space: The development of spatial representation and reasoning. Cambridge, MIT Press, 2000.

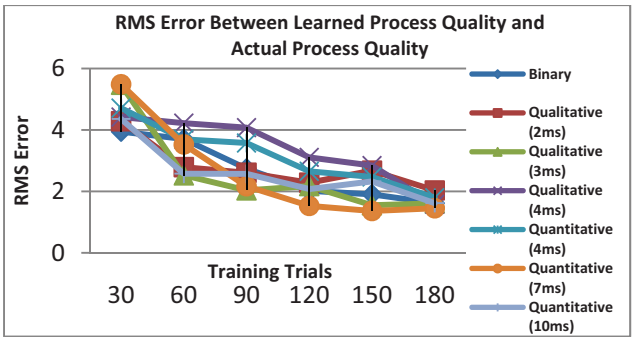

Figure 10. Depicts the RMS between learned solution quality estimates for using each deliberative process (Q) and actual solution quality generated by each process.

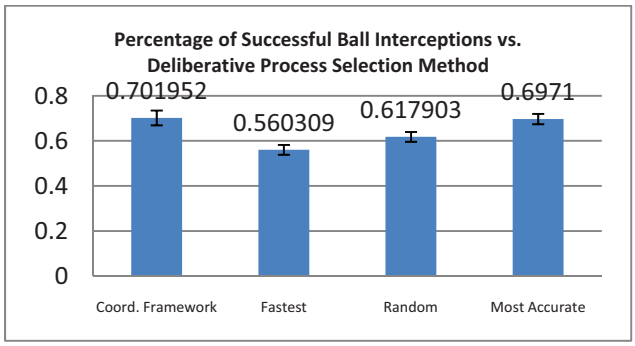

Figure 11. Task performance in terms of the percentage of successful interceptions. The control are described in Section 4C.

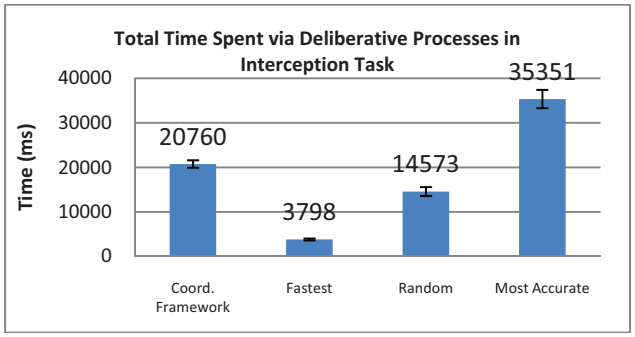

Figure 12. The average amount of time spent reasoning by each deliberative process over 100 trials. Bars indicate 95% confidence intervals

[2] Gagliolo, M. and J. Schmidhuber, "Learning dynamic algorithm portfolios." Annals of Mathematics and Artificial Intelligence 47(3-4): 295 – 328, 2006

[3]Wolpert, D, "The lack of a priori distinctions between learning algorithms." Neural Computation 8(7): 1341-1390, 1996.

[4] Rice, J.R., "The algorithm selection problem." Advances in Computers, 15:65-118, 1976.

[5] Guo, H. *Algorithm selection for sorting and probabilistic inference: A machine learning-based approach*. PhD Thesis, Kansas State University.

[6] Cox, M.T., "Metacognition in computing: a selected research review." Artificial Intelligence 169(2):104-141.

[7] Fink, E. How to solve it automatically: Selection among problem solving methods.AIPS, 1998.

[8] Leyton-brown, K., Nudelman,E., Andrew, G., Mcfadden, J., Shoham, Y. A portfolio approach to algorithm selection. IJCAI 2003.1542-1543, 2003.

[9] Arkin, R.C. Behavior based robotics. MIT Press, 1998.

[10] Kleer, J.D., Multiple representations of knowledge in a mechanics problem solver, IJCAI-77, p. 299-304, 1977.