Learning to Estimate: A Case-Based Approach to Task Execution Prediction

Bryan Auslander[1], Michael W. Floyd[1], Thomas Apker[2], Benjamin Johnson[3], Mark Roberts[3], and David W. Aha[2]

[1]Knexus Research Corporation; Springfield, VA; USA
[2]Navy Center for Applied Research in Artificial Intelligence;
Naval Research Laboratory (Code 5514); Washington, DC; USA
[3]NRC Postdoctoral Fellow; Naval Research Laboratory (Code 5514); Washington DC; USA
[1]{*first.last*}@knexusresearch.com | [2]{*first.last*}@nrl.navy.mil |[3]{*first.last.ctr*}@nrl.navy.mil

Abstract. A system that controls a team of autonomous vehicles should be able to accurately predict the expected outcomes of various subtasks. For example, this may involve estimating how well a vehicle will perform when searching a designated area. We present CBE, a case-based estimation algorithm, and apply it to the task of predicting the performance of autonomous vehicles using simulators of varying fidelity and past performance. Since there are costs to evaluating the performance in simulators (i.e., higher fidelity simulators are more computationally expensive) and in deployment (i.e., potential human injury and deployment expenses), CBE uses a variant of local linear regression to estimate values that cannot be directly evaluated, and incrementally revises its case base. We empirically evaluate CBE on Humanitarian Assistance / Disaster Relief (HA/DR) scenarios and show it to be more accurate than several baselines and more efficient than using a low fidelity simulator.

1 Introduction

Humanitarian Assistance / Disaster Relief (HA/DR) missions can occur without warning and require a rapid response to minimize damage and preserve human life. Additionally, they often occur in remote areas (e.g., an avalanche site) or dangerous locations (e.g., flooded towns, cities damaged by earthquakes, active wildfires), so it may be difficult for human relief workers to safely assist. Instead, autonomous vehicles can be used in place of, or in collaboration with, humans to allow for quicker and safer deployments.

We present Case-Based Estimator (CBE), a utility component of a larger HA/DR system that assigns autonomous vehicles to search areas in disaster zones. CBE estimates the performance of numerous vehicle-zone pairings and allows a human operator or automated mission manager to make informed decisions about how best to allocate the vehicles. Missions vary in their properties (i.e., type of disaster, location, terrain, type of vehicles, size of relief team). Thus, CBE may lack knowledge about how the autonomous vehicles will perform and must instead rely on simulators with varying fidelity. However, given the real-time nature of HA/DR missions there may not be time to evaluate every vehicle-zone pairing in every simulator. Instead, CBE will need to use information from the lower fidelity, less computationally expensive simulators to predict performance on the higher fidelity simulators and select a subset of vehicle-zone pairs to examine in more detail. This process employs regression to estimate the performance in successively higher fidelity simulators and allows the decision maker (e.g., Operator or automated mission planner) to make informed decisions on which tasks to assign to vehicles. We report an empirical study in which CBE yields more accurate results than lower fidelity simulators and outperforms unfiltered regression approaches.

In this paper we describe CBE and how it uses data from simulators (introduced in section 3) of varying fidelity to predict the performance of autonomous vehicles. Section 2 examines related work in the areas of case-based estimation and agent deployment. Section 3 describes the HA/DR domain. Section 4 briefly summarizes our HA/DR command system. Section 5 focuses on how we use CBR to estimate the performance of autonomous HA/DR vehicles. We evaluate our approach in Section 6, followed by a discussion of our results in Section 7 and concluding remarks in Section 8.

2 Related Work

Our current work focuses on online numeric prediction; we compute a linear regression equation from a subset of the most similar cases' outcomes using an online algorithm. This is an example of locally weighted regression (LWR) (Cleveland & Devlin, 1988), and in particular of algorithms that compute local estimates of the regression surface (Atkeson et al., 1997a). These popular algorithms have a long history of use in, for example, robotics control tasks (Atkeson et al., 1997b). Many variants have been examined in the CBR literature, including in the context of case-based reinforcement learning techniques (e.g., Aha & Salzberg, 1993; Gabel & Riedmiller, 2007; Molineaux et al., 2008). Given a problem p, LWR algorithms identify the set K of p's k-nearest neighbors and compute a linear or nonlinear regression equation from K's (numeric) solution values. These are often similarity-weighted, where the most similar neighbors exert more influence on the derivation of the equation. This equation is then used to predict a solution value for p. Our algorithm, CBE, computes a simple unweighted linear regression model to make predictions, but where the value of k is not fixed (it varies depending on which cases exceed a similarity threshold). We have found it to perform well in our application, and leave the investigation of other LWR methods for future work. There are also similarities to two-stage retrieval models such as MAC/FAC in Forbus et al. (1995). It uses a simple similarity metric to identify a subset of cases to evaluate with a more comprehensive structural analysis. This is similar to CBE, which uses a function based estimate of simulation performance to retrieve promising candidates for further evaluation using more rigorous simulation models.

CBR has previously been studied for robotics applications. For example, Likhachev et al. (2002) use CBR to learn parameter settings for the behavior-based control of a ground robot in environments that change over time. While they focus on motion control for a single robot, we instead focus on the high-level control of robot teams. Ros et al. (2009) focus on action selection for RoboCup soccer, and use a sophisticated representation and reasoning method. However, this body of research focuses on motion planning for relatively short-term behaviors, whereas we focus on longer duration plans that are monitored by a goal reasoning (GR) module (see Section 4).

GR agents that employ CBR techniques have been used for other control tasks, such as formulating the goals for team coordination (Jaidee et al., 2013), predicting the behavior of hostile agents (Borck et al., 2015), and recognizing the plans of an agent's teammates (Gillespie et al., 2015). However, in contrast to these other integrations, our focus is on predicting the outcomes of a plan executed by a set of robots.

In (Auslander et al., 2014) we described a CBR algorithm that sets the parameter values of complex HA/DR plans involving a heterogeneous set of unmanned autonomous vehicles that search multiple Areas of Interest (AOI). We represented cases using a similar ⟨problem, solution, outcome⟩ tuple. Our algorithm found solution parameter settings that performed well by adapting similar cases and using their outcome metrics to vote on parameter settings. When executing plans generated using our case-based algorithm on problems with high uncertainty, it outperformed plans generated using baseline approaches. In this paper, we instead focus on a complementary problem: estimating similar outcomes given a problem and solution parameter settings. These two approaches can potentially be combined in the future to improve parameter setting by estimating the performance of a proposed solution.

Finally, CBR has previously been studied for military applications, including disaster response. For example, Abi-Zeid et al. (1999) studied incident prosecution, including real time support for situation assessment in search and rescue missions. Their ASISA system uses CBR to select hierarchical information-gathering plans for situation assessment. Muñoz-Avila et al.'s (1999) HICAP instead uses conversational CBR to assist operators with refining tasks in support of noncombatant evacuation operations. SiN (Muñoz-Avila et al., 2001) is an extension that integrates a planner to automatically decompose tasks where feasible. However, while these systems use planning modules to support rescue operations, they do not predict the outcomes of a given plan's execution, nor focus on coordinating robot team behaviors.

3 Humanitarian Assistance / Disaster Relief Operations

HA/DR operations (O'Connor, 2012) are performed by several countries in response to events such as Hurricane Katrina (August 2005), the Haiti earthquake (January 2010), and Typhoon Haiyan (November 2013). Before any personnel can begin operations, information about the Area of Operations must be acquired (e.g., locations of survivors, infrastructure condition, viable ingress points, and evacuation routes). This information will also need to be continuously updated as the situation develops. Each Area of Operation is composed of one or more Areas of Interest that need to be searched.

Current operations employ remotely controlled drones and human-piloted helicopters to gather this information. We are developing methods for deploying a heterogeneous team of autonomous unmanned vehicles with appropriate sensor platforms to automate much of this process, so as to reduce time and cost. This should enable responders to perform critical tasks more quickly for HA/DR operations. Independently of which system is used, an Operator given a list of missions must be able to prioritize which missions should be planned for and scheduled.

We focus on a method for comparing potential mission outcomes to enable the Operator or mission planner to select which missions to perform. These missions can be automatically generated from our goal reasoning system or provided by operators. This module's task is to provide estimates of the outcome metrics, which can be used to make more informed decisions on what to dispatch. This may yield better plans.

We use three simulations of varying fidelity in the CBE: an inexpensive function-based approach, a quick low fidelity simulator, and a slower high fidelity simulation. The first can estimate a metric without simulation (e.g., by computing the path a vehicle might take and dividing the path length by the vehicle's speed to estimate time required). To ensure efficiency, these estimates do not account for important factors such as wind and fuel levels, but they do provide instant, initial results.

Our low fidelity simulation is MASON (Luke et al., 2005), a discrete-event multi-agent simulator that models all physics behaviors and physicomimetics control (Martinson et al., 2011). MASON models the physical movements of generic agents acting in the environment. However, it lacks specific physical models of its actors and does not account for detailed problem factors such as the effects of wind. MASON's low fidelity allows it to more quickly generate results, but these are likely to be less accurate because it does not model all features.

Open AMASE (Duquette, 2009) is the highest fidelity simulation we use, and in this paper we use it as a substitute for a real-world environment. This simulation models small tactical unmanned aircraft systems (STUAS) using a kinematic flight dynamics model that includes environmental effects (e.g., wind) on performance. AMASE also has facilities for modelling the field of view of cameras mounted on the STUAS based on the vehicles' six degree of freedom pose. This allows AMASE to calculate a metric for coverage defined as the area the sensor observed at a specified resolution. The lower fidelity models cannot produce this metric, and instead assume the paths followed produced full coverage. Figure 1 displays an example problem using both real-world data and AMASE's representation.

4 Situated Decision Process

To intelligently act in domains like HA/DR, a team of autonomous agents must continually monitor, evaluate, and dispatch new tasks or goals. To this end, we have designed a system architecture called the *Situated Decision Process* (SDP) (Roberts et al., 2015). In the SDP, a centralized Mission Manager subsystem assigns primitive goals to teams of autonomous agents, based on the input of an Operator and the vehicles' observations during execution. Intelligent, autonomous evaluation and selections of goals or tasks during execution requires rapid, accurate estimation of multiple scenario parameters.

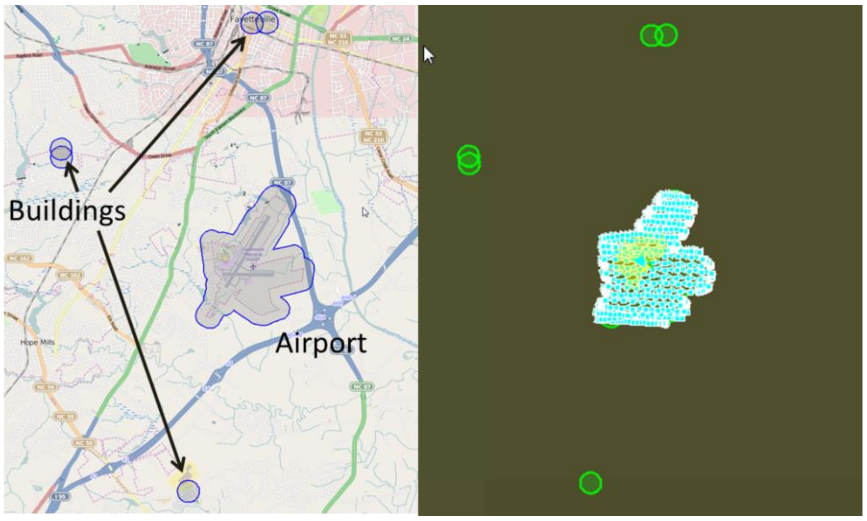

Figure 1: Left: Representation of a problem set using OpenStreetMap data. Right: Same problem shown in AMASE with tracks for the airport region.

In HA/DR scenarios, the vehicles must quickly react to changes in the perceived environment, as well as to changes to the Operator's inputs. Doing so requires the rapid evaluation of such changes; it requires the ability to predict the effect of performing tasks more quickly than can be simulated with high fidelity and with more accuracy than can be achieved with low fidelity. This led us to consider using CBR to quickly and accurately estimate the parameter settings used by the Mission Manager to intelligently evaluate the utility of the vehicles' goals and tasks.

At the individual vehicle and sub-team level, CBE's estimates can be used in *motivators* for goal selection in a goal reasoning algorithm (Wilson et al., 2013). This would help us implement the situated portion of the SDP by permitting decision making on vehicles without direct access to the Mission Manager. This enables vehicles to choose predictable actions that should provide locally optimal results.

5 Case-Based Performance Estimation

To provide the data necessary for the Mission Manager to make informed decisions about its various vehicle deployment options, we use the CBE to evaluate mission options in HA/DR scenarios. We describe its case representation in Section 5.1 and case similarity metric in Section 5.2. The CBR algorithm and knowledge acquisition technique are presented in Section 5.3.

Table 1: CBE's Case Representation

Case Component		Attribute Name	Description
Problem (p)	Problem Description	Total Area of AOI	Total area of AOI in m^2
		Distance to AOI	Distance from vehicle to AOI in m
		Wind Speed	The speed of wind in m/s
		Wind Direction	Wind angle relative to AOI orientation
	Solution	STUAS Configuration	
Outcomes (O_{all})	O_e	Duration	Time to complete operation
		Energy	Amount of energy consumed in Joules
	O_l $O_{l_{estimate}}$	Duration	Time to complete operation
		Energy	Amount of energy consumed in Joules
	O_h $O_{h_{estimate}}$	Duration	Time to complete operation
		Coverage	Percentage of area observed by sensor
		Energy	Amount of energy consumed in Joules

5.1 Case Representation

We represent a case $C = \langle p, O_{all} \rangle$ as a problem p and the set of all outcomes O_{all} when that problem is evaluated using models of different fidelity. In this paper, we use three models of increasing complexity: an evaluation function, a low fidelity simulator (MASON), and a high fidelity simulator (AMASE). Similarly, we are also interested in estimating the performance when using the simulators (e.g., if the simulator is unavailable or computationally expensive). As such, the case contains the outcomes generated by the evaluation function (O_e), the low fidelity simulator (O_l), and the high fidelity simulator (O_h), and estimates of the low and high fidelity simulations ($O_{l_{estimate}}$ and $O_{h_{estimate}}$): $C = \langle p, O_e, O_l, O_{l_{estimate}}, O_h, O_{h_{estimate}} \rangle$.

Table 1 provides detail on this representation. A problem p is composed of a problem description and proposed solution. The problem description is further divided into four features that characterize an aerial search task. *Total Area of AOI* is the total size of the area of interest (AOI) (i.e., the area being searched) in square meters. The *Distance to the AOI* is a measure of how far the search vehicle would have to travel to reach the center of the area. This becomes important as the trip time becomes a significant cost of the operation. *Wind Speed* is a measure of the magnitude of the wind in meters per second. Wind is a large source of error between the low and high fidelity simulations and tracking it enables a system to separate cases by the wind magnitude. *Wind Direction* is a measure of the alignment of the wind relative to the search area; it is a value in [0°, 90°].

A solution represents the configuration of the search vehicles that will be assigned to the search area (e.g., vehicle types, the number of vehicles, sensor configurations). Here we focus on problems where a single vehicle of a fixed type is assigned to perform the search. (See (Auslander et al., 2014) for more complex solutions.)

Initially, each case contains only the problem description with unknown values for each outcome. As more information is obtained (i.e., evaluating the problem using the evaluation function or one of the simulators, or estimating the outcomes), it is added to the case. Only promising problems identified from the performance estimates are evaluated at the higher, more computationally expensive fidelities, so not all cases will have values for all outcomes. All outcomes have measurements for the search *duration* (seconds) and search *energy* (joules), while search *coverage* (percent of area observed with sensors) can be measured only by the high fidelity simulator and is therefore only contained in its outcomes. The estimated values may be continually overwritten, if new data becomes available that modifies these values, while the data obtained from simulation is recorded only once. The estimates serve as an inexpensive temporary measurement until the actual simulation is run; they are no longer used after the actual values are known.

5.2 Case Similarity

Case similarity is calculated using a weighted comparison of the problem features in two problems. Given two problems p_1 and p_2, the similarity metric (Equation 1) calculates a similarity between 0 and 1. Each problem contains n features, and each feature f_i is given a weight w_i (*maxValue(i)* and *minValue(i)* represent the maximum and minimum value the i^{th} feature can take). In CBE, there are four problem features: *Total Area of AOI*, *Distance to AOI*, *Wind Speed*, and *Wind Direction*. *Total Area of AOI* and *Distance to AOI* are assigned weights of 2.0, whereas other features are assigned weights of 1.0 to enable better separation of regions of varying sizes and locations. A weighted approach is used to allow more flexibility in discriminating among cases (e.g., emphasizing the geometric properties of the domain).

$$sim(p_1, p_2) = \frac{1}{\sum_{i=1}^{n} w_i} \sum_{i=1}^{n} w_i \left(1 - \frac{|p_1.f_i - p_2.f_i|}{\text{maxValue}(i) - \min Value(i)}\right) \quad (1)$$

5.3 Performance Estimation Algorithm

CBE (Algorithm 1) enables a computationally inexpensive and accurate evaluation of potential configurations provided by the Mission Manager. Evaluating each potential configuration in the simulators can be expensive. Thus, CBE allows the Mission Manager to provide feedback about which configurations should be evaluated in more detail. To begin with, CBE receives a set of problems $Probs$ representing possible missions under consideration from the Mission Manager (MM). For each problem $p \in Probs$, it retrieves a similar case C from case base CB using $SimilarCase(p, \lambda, CB)$, which examines all cases in CB that are above similarity λ to p using from equation 1 and returns the most similar case with a known O_h (i.e., preference is given to cases with more known values). If no above-threshold cases have a known O_h, the most similar case is returned. If no cases are above threshold, a null value is returned. If a case is retrieved, CBE uses it. Otherwise, CBE evaluates p using the evaluation function (i.e., computing O_e) and creates a new case (the values for all other outcomes are set to null). The retrieved or created case is then added to a set of cases to be further evaluated.

Algorithm 1: Case-Based Estimator (CBE)

```
Inputs: $probs = \{p_1, p_2, \ldots, p_n\}$
Returns: $CB'_E$ // Performance of problems in MM filtered subset of cases
Legend:
    $CB$              // The (entire) case base
    $CB_E \leftarrow \{\emptyset\}$  // Subset of cases to be evaluated

Function: $FindBestProblem(probs)$ returns $CB'_E$
foreach $p \in probs$ do
    $C \leftarrow SimilarCase(p, \lambda, CB)$;
    if $C = \emptyset$ then
        $O_e \leftarrow EvaluationFunction(p)$;
        $C_{new} \leftarrow \langle p, O_e, \emptyset, \emptyset, \emptyset, \emptyset \rangle$;
        $CB \leftarrow CB \cup C_{new}$;
        $CB_E \leftarrow CB_E \cup C_{new}$;
    else
        $CB_E \leftarrow CB_E \cup C$;
$CB_E \leftarrow ComputeEstimates(CB_E, CB)$;
$CB'_E \leftarrow MissionManagerFilter(CB_E)$;
$CB'_E \leftarrow RunMASONAndReviseCases(CB'_E)$;
$CB'_E \leftarrow ComputeEstimates(CB'_E, CB)$;
return $CB'_E$;

Function: $ComputeEstimates(CB_E, CB)$
foreach $C \in CB_E$ do
    if $C.O_l = \emptyset$ then
        $C.O_{l_{estimate}} \leftarrow EstimateMASON(C, CB)$;
    if $C.O_h = \emptyset$ then
        $C.O_{h_{estimate}} \leftarrow EstimateAMASE(C, CB)$;
return $CB_E$
```

If the problem's MASON and AMASE values are not known (i.e., the problem has never been evaluated in the simulators), CBE then estimates the MASON and AMASE values (i.e., $O_{l_{estimate}}$ and $O_{h_{estimate}}$). The resulting cases are then sent to the Mission Manager for filtering because it is best able to choose what problems and metrics to optimize over given the overall mission context.

The Mission Manager returns a subset of cases (CB'_E) for further evaluation. For each of these cases, if the actual MASON outcome values are not known, it is run in the MASON simulator and its corresponding O_l values are revised. Afterward, the estimation routine is run again to generate new estimations for the AMASE outcome values and the resulting subset is returned to the Mission Manager. Not shown in

Algorithm 1 is once the Mission Manager has filtered the set of problems a subset of these are picked to be deployed based on the Mission Manager's criteria. The resulting AMASE outcome values are subsequently stored in the case (i.e., as O_h) if the case did not previously have AMASE outcome values. If AMASE outcome values already exist (e.g., for a repeated surveillance task), rather than ignore the data a new case is created from the current problem. Its MASON outcome values are also computed to ensure no cases have AMASE outcome values without MASON outcome values.

The functions $EstimateMason(C, CB)$ and $EstimateAMASE(C, CB)$ are implemented using a linear regression algorithm for each outcome attribute. For MASON, the linear regression function takes the form $(p, O_e) \rightarrow O_{l_{estimate}}$. Similarly, the AMASE regression function is of the form $(p, O_l) \rightarrow O_{h_{estimate}}$. These regression algorithms are trained using cases that are above similarity θ to the current case and have known O_l (for MASON regression) or O_h (for AMASE regression) values. This similarity threshold ensures the regression functions are generated using only data from similar problems, helping isolate problems into clusters.

This is an online learning algorithm for estimation, with data acquired every time the estimation system is run. For each problem, a new case can potentially be generated. As the Operator selects problems to evaluate further, the MASON and AMASE outcome metrics are added to the cases. As more values are known, the algorithm will have more data to use for regression and should increase estimation accuracy.

6 Empirical Study

We empirically tested the following hypotheses:

H1: CBE's estimate of a problem's outcome approaches the actual outcome when evaluated using the high fidelity simulation over time.

H2: CBE provides more accurate estimates than the evaluation function and low fidelity simulator.

H3: CBE is more computationally efficient than the low fidelity simulator as the number of cases increases.

H4: CBE's filtered regression approach yields more accurate predictions than a non-filtered regression.

In the following sections we describe the evaluation methods, algorithms tested and metrics used.

6.1 Empirical Method

An objective of these tests is to verify that CBE accurately predicts the performance of a configuration when run on a high fidelity simulation. Thus, our ideal performance baseline is provided by the high-fidelity AMASE.

Problem sets were generated using a custom PostGIS system (Roberts et al., 2015). Each problem was formed by choosing a random airport from the OpenStreetMaps data set (Geofabrik, 2014) and finding five random buildings within a 3-5 kilometers radius of the airport. Each of these six locations is given a buffer region of 300 meters around their perimeters and the result is the search area to use in a given problem. Each search area is also assigned a random wind speed between 0-20 meters per second. The six search areas (i.e., airport and five nearby buildings) are stored as a search and rescue (S&R) problem. We repeated this 100 times to obtain 100 S&R problems. Problem features (e.g., Distance to AOI, Wind Direction) are derived from these problems at run time.

Each S&R problem was used to create a problem set that contains potential vehicle assignments for the problem. In CBE, only one vehicle assignment was used (i.e., the STUAS with default camera configuration). Future work will evaluate other vehicles and configurations as parameters, such as the use of static cameras and multiple vehicles. All problems in the problem set were run in AMASE to obtain ground truth data (i.e., how well that vehicle will perform when assigned to search a specific region).

A problem set run consists of giving an entire problem set to CBE and comparing its estimates to the known ground truth. Because we cannot know what the Mission Manager seeks at this level of abstraction, since it could be a human or intelligent subsystem, our method for selecting a subset of cases to run on MASON randomly selects 4 of the 6 cases. Similarly, when the final estimates are returned, 2 randomly selected problems (among the 4 selected) will be evaluated in AMASE.

A test run consists of randomly ordering the 100 problem sets and sequentially giving them as input to CBE, simulating 100 sequential uses of CBE. At the end of the 100 runs there will be 200 fully evaluated cases. We repeated this process 50 times and aggregated the results. All regression calculations were computed using WEKA's linear regression implementation (Hall et al., 2009).

6.2 Algorithms and Baselines Tested

We used the following algorithms and baselines to evaluate CBE. Each was run using the same evaluation problems presented in identical orderings.

- **CBE:** We set $\theta = 0.75$ to allow discrimination between building searches and airport searches. We set $\lambda = 0.99$ so that cases are reused only when they are highly similar to the problem.
- **Func**: Results obtained from running the estimation function on each problem.
- **MASON:** Results obtained by running MASON on each problem.
- **FuncReg:** Results obtained using linear regression to estimate AMASE's outcomes using problem features and the function estimate. This uses all available AMASE data for regression (i.e., not only data from similar problems, as with the CBR approach).
- **MASONReg:** Similar to *FuncReg*, but problem features and MASON outcomes are used to estimate AMASE outcomes. This also uses all available AMASE data for regression.

FuncReg and *MASONReg* use all data that is available to perform linear regression. For example, for the 100th input problem set 198 cases are used (since two AMASE outcomes are determined from each of the previous 99 input sets) while the 1st problem set will have no known AMASE outcomes. When any algorithm is unable to predict an AMASE outcome (i.e., no data to perform regression) a default error of 200% is used. The *Func* and *MASON* baselines serve to show that using the values from only the lower fidelity simulators is inferior to using a mapping function such as the regression approach of CBE.

6.3 Results and Analysis

We now describe whether our results support our hypotheses:

H1: Figure 2 displays results showing support for H1. It graphs the mean error of CBE over 50 runs after it has calculated its k^{th} AMASE outcome estimate (100 problem sets with two estimates per set). As the number of problems evaluated increases, the error decreases and eventually converges to approximately20% error, which is an improvement over our low fidelity estimate as shown in H2. The graph shows the error when predicting Duration. Although not shown, Energy converges similarly.

H2: Figure 2 also displays the performance of baselines *MASON* and *Func*. CBE consistently outperforms the evaluation function and eventually outperforms *MASON* giving support to H2. Table 2 confirms this; using a paired *t*-test we found that CBE significantly outperforms *Func* for Energy and Duration. It also significantly outperforms *MASON* overall for Energy and for Duration over the final 75% of problems (i.e., after learning).

H3: CBE requires fewer problems to be evaluated than if every problem is evaluated in MASON (only 4 of 6 are evaluated in MASON, so 67% of the evaluations). Reducing necessary simulations runs is a large reduction in run time considering that, for an example run of an 18 minute real world mission, the MASON simulation can take 35 seconds while AMASE takes 95 seconds. Additionally, setting an appropriate value for λ can influence how often MASON is used by CBE. If there is a case that is similar to an input problem, that problem does not need to be evaluated in MASON if the case has a recorded MASON outcome. In the current evaluation about 23 cases per test run were found to be similar enough to an input problem to be reused. If the Mission Manager wanted evaluations for all of those problems (i.e., they were among the 4 of 6 selected for MASON evaluation), that would result in 23 fewer MASON evaluations out of 400. However, the Mission Manager may not require evaluations for any of those problems and would instead evaluate other problems, resulting in no additional improvements. Reducing λ to 0.98 increases this to 107 reuses. This hypothesis has some support, but further exploration of parameters is warranted to find optimal values for this domain.

H4: Support of H4 is shown in Figure 3 which graphs the results of CBE versus the two full regression approaches (i.e., they use *all* the cases), which begin with no data for the first two AMASE estimates and as such default to 200% error. This accounts for the highest error in the first two AMASE estimates. In contrast, CBE uses the estimation function and MASON estimates, accounting for a lower initial error. For the remaining AMASE estimates, all three algorithms converge towards approximately 20% error. Although CBE does not appear to converge faster, its errors are never as large as the initial regression models. This could be due to case reuse or the filtering of non-similar cases in the regression calculation.

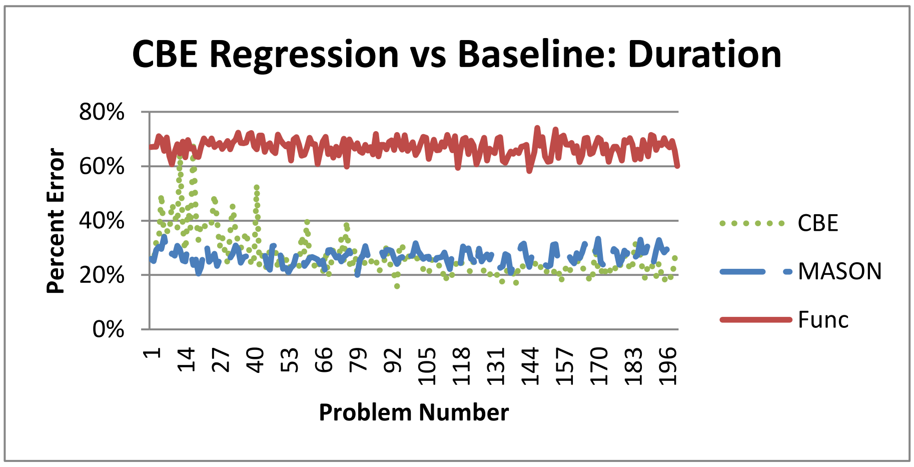

Figure 2: Graph plotting percent error for CBE and the baseline algorithms across the 200 problems on 50 runs.

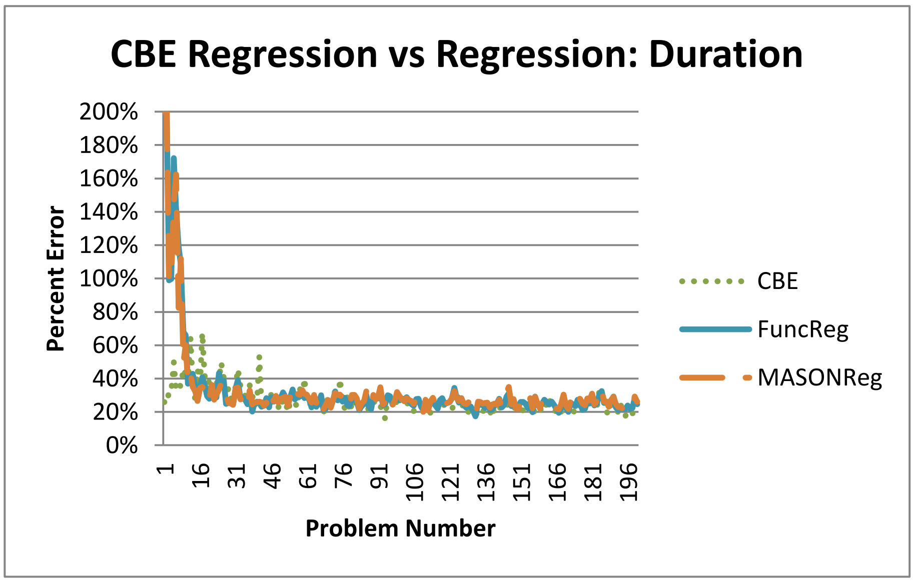

Figure 3: The error of CBE and the regression algorithms for 200 problems over 50 runs.

Shown in Figure 3, CBE's performance is better than or equal to the other algorithms. In the comparison of the 50 test run averages, CBE significantly outperformed both alternatives using a paired *t*-test; see Table 2. Table 3 displays the mean number of times, across all 50 runs, that CBE recorded lower error than the other algorithms.

Additionally, it shows the mean reduction in absolute error when using CBE across all test runs.

Table 2: Results of *t*-tests ($p <$ value) showing CBE's improvement vs. other algorithms

CBE vs	Duration	Energy
Func	0.00000	0.00000
MASON	0.34228 (0.0 after 25% of the cases)	0.00000
FuncReg	0.00137	0.00446
MASONReg	0.00062	0.00130

Table 3: Improvement of CBE versus the regression algorithms

	Duration		Energy	
CBE vs	**# Improvements**	**Mean Error Reduction**	**# Improvements**	**Mean Error Reduction**
FuncReg	54.69%	5.05%	54.05%	2.75%
MASONReg	55.09%	5.33%	55.03%	2.97%

7 Discussion

The results in Section 6 clearly indicate the benefits of CBE, which recorded a 5% reduction in Duration error and an almost 3% reduction in Energy error. The reason for Energy's lower improvement could be differences with how the low and high fidelity simulators are modelling recharging. In MASON a vehicle is supposed to remain still while recharging, while AMASE (which was built to model fixed wing aircraft) does not restrict movement as much while recharging. Future versions of these simulators will address these discrepancies and also implement a procedure for returning to base and landing to increase scenario realism.

We expect that further improvements to performance will be found as more discriminating problem features are identified. For example, another type of vehicle would yield entirely new data clusters as Energy burn rates, and flight profiles would differ. As more data is collected over time the accuracy of the algorithms should increase and require fewer simulation runs.

8 Conclusion

Most CBR systems that estimate functions, such as cost, attempt to find a similar case and adapt their solution. We report on a novel online hybrid algorithm that can reuse prior learned values from similar problems and creates new estimates for others. For the scenarios in the domain we examine, the Case-Based Estimator (CBE) produced more accurate estimates from less data than two other regression algorithms.

One of the next steps from these results is to combine the benefits of this approach with the parameter selection approach from our previous investigation (Auslander et al., 2014). Benefits may include improving suggested solutions by estimating their actual outcomes. Beyond this there are many ways to improve the CBE algorithm.

One of the most promising directions would be the exploration of non-linear regression models. It is likely that some of our problem features are not independent, and a model that considers co-variance information may return more accurate results. We expect there to be tradeoffs in performance with these new models (e.g., additional computational resources required for increased training samples).

Over time the amount of data in the case base will become sufficiently large to necessitate the use of case-base maintenance techniques. While in general the accuracy of regression algorithms will increase given more data, improvements may also accrue by removing anomalous cases. In addition, as more data is obtained it may be possible to be more discriminative in case selection by increasing λ and θ. Future research may identify ways to scale these parameters with the data.

Acknowledgements

Thanks to OSD ASD (R&E) for sponsoring this research.

References

Abi-Zeid, I., Yang, Q., & Lamontagne, L. (1999). Is CBR applicable to the coordination of search and rescue operations? A feasibility study. *Proceedings of the Third International Conference on Case-Based Reasoning* (pp. 358-371). Seeon, Germany, Springer.

Aha, D.W., & Salzberg, S.L. (1993). Learning to catch: Applying nearest neighbor algorithms to dynamic control tasks. *Proceedings of the Fourth International Workshop on Artificial Intelligence and Statistics* (pp. 363-368). Ft. Lauderdale, FL: Unpublished.

Atkeson, C., Moore, A., & Schaal, S. (1997a). Locally weighted learning. *Artificial Intelligence Review*, *11*(1-5), 11-73.

Atkeson, C., Moore, A., & Schaal, S. (1997b). Locally weighted learning for control. *Artificial Intelligence Review*, *11*(1-5), 75-113.

Auslander, B., Apker, T., & Aha, D.W. (2014). Case-based parameter selection for plans: Coordinating autonomous vehicle teams. *Proceedings of the Twenty-Second International Conference on Case-Based Reasoning* (pp. 32-47). Cork, Ireland: Springer.

Borck, H., Karneeb, J., Alford, R., & Aha, D.W. (2015). Case-based behavior recognition in beyond visual range air combat. *Proceedings of the Twenty-Eighth Florida Artificial Intelligence Research Society Conference* (pp. 379-384) . Hollywood, FL: AAAI Press.

Cleveland, W. S., & Devlin, S. J. (1988). Locally weighted regression: an approach to regression analysis by local fitting. *Journal of the American Statistical Association*, *83*(403), 596-610.

Duquette, M. (2009). Effects-level models for UAV simulation. In *Proceedings of the AIAA Modeling and Simulation Technologies Conference*. Chicago, IL: AIAA.

Forbus, K. D., Gentner, D. and Law, K. (1995). MAC/FAC: A model of similarity-based retrieval. *Cognitive Science, 19*, 141–205.

Gabel, T., & Riedmiller, M. (2007). An analysis of case-based value function approximation by approximating state transition graphs. *Proceedings of the Seventh International Conference on Case-Based Reasoning* (pp. 344-358). Belfast, Northern Ireland: Springer.

Geofabrik (2014). OpenStreetMap data extracts. Accessed from http://download.geofabrik.de/index.html.

Gillespie, K., Molineaux, M., Floyd, M.W., Vattam, S.S., & Aha, D.W. (2015). Goal reasoning for an autonomous squad member. In D.W. Aha (Ed.) *Goal Reasoning: Papers from the ACS Workshop*. Atlanta, GA: [www.cc.gatech.edu/~svattam/goal-reasoning].

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., & Witten, I.H. (2009). The WEKA data mining software: An update. *SIGKDD Explorations*, *11*(1), 10-18.

Jaidee, U., Munoz-Avila, H., & Aha, D.W. (2013). Case-based goal-driven coordination of multiple learning agents. *Proceedings of the Twenty-First International Conference on Case-Based Reasoning* (pp. 164-178). Saratoga Springs, NY: Springer.

Likhachev, M., Kaess, M., & Arkin, R. (2002). Learning behavioral parameterization using spatio-temporal case-based reasoning. *Proceedings of the International Conference on Robotics and Automation* (pp. 1282-1289). Washington, DC: IEEE Press.

Luke, S., Cioffi-Revilla, C., Panait, L., Sullivan, K., & Balan, G. (2005). MASON: A multiagent simulation environment. *Simulation, 81*(7), 517–527.

Martinson, E., Apker, T., & Bugajska, M. (2011). Optimizing a reconfigurable robotic microphone array. *Proceedings of the International Conference on Intelligent Robots and Systems* (pp. 125-130). San Francisco, CA: IEEE Press.

Molineaux, M., Aha, D.W., & Moore, P. (2008). Learning continuous action models in a real-time strategy environment. *Proceedings of the Twenty-First Florida Artificial Intelligence Research Conference* (pp. 257-262). Coconut Grove, FL: AAAI Press.

Muñoz-Avila, H., Aha, D., Breslow, L., & Nau, D. (1999). HICAP: An interactive case-based planning architecture and its application to noncombatant evacuation operations. *Proceedings of the Ninth National Conference on Innovative Applications of Artificial Intelligence* (pp. 879-885). Orlando, FL: AAAI Press.

Muñoz-Avila, H., Aha, D., Nau, D., Weber, R., Breslow, L., & Yaman, F. (2001). SiN: Integrating case-based reasoning with task decomposition. *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence* (pp. 999-1004). Seattle, WA: Morgan Kaufmann.

O'Connor, C. (2012). Foreign humanitarian assistance and disaster-relief operations: Lessons learned and best practices. *Naval War College Review*, *65*(1), 152-160.

Roberts, M., Apker, T., Johnson, B., Auslander, B., Wellman, B., & Aha, D.W. (2015). Coordinating robot teams for disaster relief. *Proceedings of the Twenty-Eighth Florida Artificial Intelligence Research Society Conference (pp. 366-371).* Hollywood, FL: AAAI Press.

Wilson, M., Molineaux, M., & Aha, D.W. (2013). Domain-independent heuristics for goal formulation. *Proceedings of the Twenty-Sixth Florida Artificial Intelligence Research Society Conference* (pp. 160-165). St. Pete Beach, FL: AAAI Press.