INTELLIGENT SYSTEMS APPLIED TO WORKLOAD MONITORING AND TASK ALLOCATION

Scott Grigsby, Jacob Crossman, Rich Frederikson, Ben Purman, and Dylan Schmorrow
Soar Technology, Inc.
Ann Arbor, MI

We have implemented an integrated system, called Computational Situation Awareness (CSA), into a prototype control station to coordinate dynamic task sharing and allow UxV operators to be quickly brought up to speed with sufficient situation awareness to effectively understand and handle new tasking. Within CSA we have implemented a real-time workload model for estimating workload across a user's visual, tactile, and cognitive resources. This workload estimate enables the system to recognize when tasks should be handed off and to whom they should be handed. By tracking user tasks, user workload, and system state, the system builds an understanding of the team's effectiveness and current capabilities and tracks a task's progress within the team. This understanding allows the system to determine which team member can best perform each task and to determine the information each operator will need to obtain or maintain SA without unduly increasing workload.

Team coordination during unmanned systems (UxS) operations is relatively simple when an intelligence, surveillance, reconnaissance (ISR) mission is proceeding normally; however, when something goes wrong (e.g., vehicle control issue) or an interesting event occurs (e.g., new contact or threat), individual human team members become bogged down and can quickly be overwhelmed (Salas, Rosen, Burke, Nicholson, & Howse, 2007). **Dynamic task sharing** – the ability to quickly re-assign tasks and responsibilities - vastly improves team coordination. Task assignment can be traded off between the UV autonomy and multiple operators based on mission context.

Two elements are required for effective task sharing and handoff: **workload assessment** and **situation awareness** management. Workload assessment enables the system to recognize when tasks should be handed off, and to whom they should be handed (Breslow, Gartenburg, McCurry, & Trafton, 2014) (Solovey, Zec, Garcia-Perez, Reimer, & Mehler, 2014). SoarTech has developed a workload model for assessing real-time workload across a user's visual, tactile, and cognitive resources and has applied this model to the problem of automated offloading of tasks from one user to another.

Approach

The basic architecture for workload estimation is illustrated in Figure 1. Events and state data (most importantly, tasks) are pulled from the internal operating picture (IOP). For each event and task, the system looks up the workload model for that event and reads the expected instantaneous workload value for that event at the given time. The basic workload assessment capability uses a 3-dimensional model to estimate workload in the cognitive, visual, and manual dimensions (these three dimensions are taken from Wickens (2008), but here the auditory mode is not incorporated to simplify the model for initial implementation).

The equation used to compute the overall workload is as follows:

$$W(t) = \sum_{i=1}^{N} \boldsymbol{max}_{k=1,2,3}\{w_k(t-t_i)\} + g(t) \tag{1}$$

$$g(t) = 0.1 \times 1.25^N \tag{2}$$

Here W(t) is the estimated workload computed as the accumulation of the maximum of the component (e.g., cognitive, visual, manual) workloads for N events/tasks. Component workloads are computed as functions of time, where each t_i is the start time for the ith event or task. The $g(t)$ component approximates the non-linear effects caused by multiple simultaneous active events (attention switching). Its constant values were tuned during laboratory tests such that workload values approximated to 1.0 when a user appeared to be overloaded. g(t) presents some problems as it scales exponentially over the whole range of N, which is not realistic and can overestimate the workload for high Ns. It works reasonably well for N < 15, but provides excessive estimates. A future version will likely replace the exponential with a sigmoid function to model a diminishing effect as N increases to large numbers.

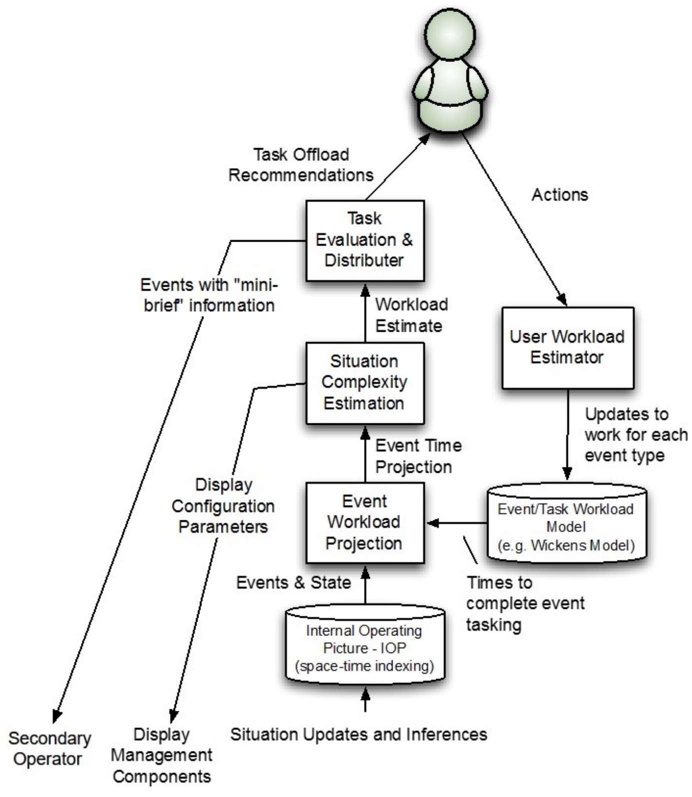
Task Offload Recommendations
Actions
Task Evaluation & Distributer
Events with "mini-brief" information
Workload Estimate
User Workload Estimator
Situation Complexity Estimation
Updates to work for each event type
Event Time Projection
Display Configuration Parameters
Event Workload Projection
Event/Task Workload Model (e.g. Wickens Model)
Events & State
Times to complete event tasking
Internal Operating Picture - IOP (space-time indexing)
Secondary Operator
Display Management Components
Situation Updates and Inferences

Figure 1. Workload estimation architecture.

This model is similar in concept to the Wickens' (2008) model, but differs in that it scales to an arbitrary number of tasks and events and accounts for change in workload over time such as temporal delays, workload ramp up, and workload ramp down. Our model is also designed to be more precise in that it seeks to derive a number that can be compared to the user's full load level (e.g., 1.0 = fully loaded).

The most important elements of the workload calculation are the workload models associated with each event and task. Figure 2 illustrates the workload model concept.

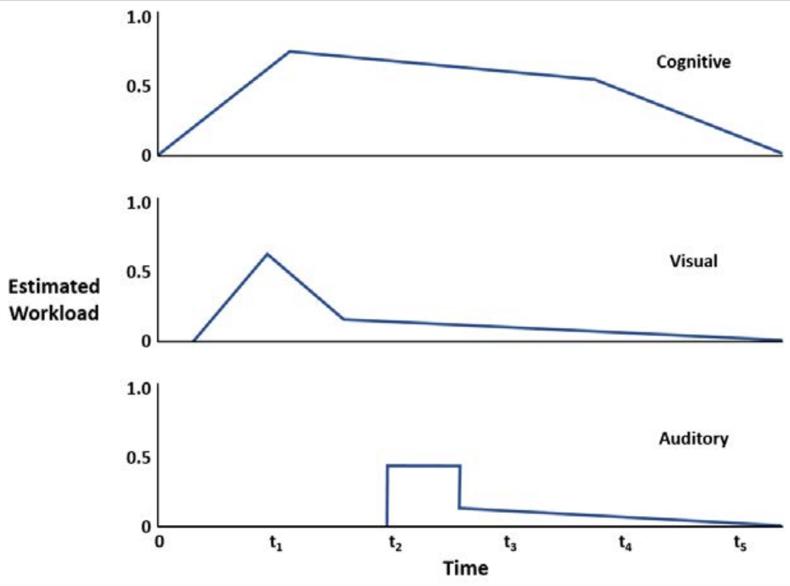

Figure 2. Sample time courses of workload estimates for a single event/task.

For each event, we define three curves, where the x-axis is time and the y-axis values are the estimated workload along the given dimension, at the given time. Each of these curves is defined with the start of the event defined as x = 0. For our model, we defined these curves as piecewise linear functions with four linear regions: (1) the delay time (time to start the effect), (2) the ramp up time (time over which the effect rises to its maximum value, (3) the peak duration, and (4) the ramp down time. Workload profiles were pre-defined for all events and all tasks within the scenario.

The expected instantaneous workload values are aggregated, using Equation 1 (abaove), to form an overall estimated workload. This value is fed into the task evaluation and distribution function where decisions on task distribution are made. When the workload is estimated to be above a single operator's capacity to execute effectively, events and tasks are offloaded to another operator. However, the decision when to offload is not straightforward. We found in our discussions with users during the evaluation that there are several ways tasks can be divided within a team.

1. They can be broken up purely based on their **timing** (e.g., round robin distribution, or secondary user gets all tasks after the main user becomes overloaded).
2. They can be broken up based on **load balancing** (e.g., the choice of which user gets which tasks is made so as to minimize user workload differences).
3. They can be made based on **geography** (e.g., each user takes care of a geographic segment)
4. They can be made based on **asset ownership** (e.g., each user gets events and tasks associated with the assets that he/she owns).

In practice, during our evaluation, operators used more than one of these approaches when they were given a choice.

We only had time and resources to select one offloading method for our initial implementation. We selected a variant of (2) for its simplicity. The variation was that the algorithm tries to load users one at a time – meaning it will not balance the load until at least one user is fully allocated. The idea behind this strategy is that it allows the secondary operator(s) to focus on other tasks while the primary operator is able to do the task alone. The algorithm is as follows:

```
1. Assign all unassigned events to the user with the highest workload (this is the primary user).
2. If this user’s workload estimate is < 1.2 (combined) then stop, otherwise --
3. Find the maximum workload event assigned to the high workload user that has occurred in the last 30
   seconds. (i.e. offload the event that is the biggest component of the workload.
4. Assign this event to the lower workload user if (event.workload < 0.9 ( high_user.workload -
   low_user.workload ))
```

We note a couple of key points regarding this algorithm. First, events/tasks are not offloaded until the workload level reaches 1.2. This allows for the lack of precision in the workload model numbers. They are not intended to be correct to a single decimal place. Second, it moves tasks to other users from largest to smallest. The idea is to take the major focus tasks away from the primary user, so the primary user can continue to maintain global SA. In the future, task allocation will incorporate user role and other factors to best determine which tasks should be distributed. It may be that a task needs to stay with an already high-workload operator simply because it would lead to the best opportunity for mission success. These aspects would need to be incorporated into the world model of the task allocation system.

Evaluation

The workload and offloading evaluation was part of a bigger evaluation study looking at operator effectiveness and situation awareness within a multi-UxS scenario. The scenario had both a Primary and a Secondary operator handling the various tasking. The evaluation was executed as a comparative study using a single control (or base) case and a single evaluation case. Both cases used the same scenario design and simulation and the same core user interface (RaptorX). Communication between operators occurred only via a chat window. The independent variable for the study and the difference between the two cases was presence or absence of the workload estimation and offloading capabilities (which were presented to the user as extensions of the Raptor X capability).

Overall, the operators were responsible for monitoring activity at a virtual port. They were given initially 4, and as the mission progressed as many as 6 UAVs to execute the mission. These assets could be used to obtain tracks of vessels and ground vehicles in and around the port. Each track was associated with a set of property data (e.g., location, size, cargo), the completeness of which is dependent on the quality of the sensor reading on the vehicle.

To achieve the mission objectives the user could do two primary things: command UAVs and inspect/update information about tracks. The user could also execute supporting actions that help make decisions and commands easier. In addition to the main tasks, various situations can arise that require operator attention (communication failure, low fuel, navigation failure, air space breach, etc.). In many cases the operator has freedom in the details of the action taken. During the course of the scenario, tasks would be offloaded automatically per the workload and task allocation algorithms but also the Primary operator could manually offload tasks to the secondary operator when desired.

Results

An interesting aspect of the evaluation was that when using the system, users did not consciously notice that automated task offloading was occurring nor did they subjectively report decreased workload. Nevertheless, it influenced how they operated and allowed them to perform more efficiently due to how it affected asset and event ordering and display in the user interface. For example, while objective measures of mission performance (time to identify targets, target tracking, etc.) showed moderate improvement, we found that the primary operator's interactions with the system was significantly reduced when task allocation was implemented from an average of 375 down to 240 ($p<0.05$, 2-tailed T-test) and, at the same time, the secondary operator's interactions stayed roughly the same (215 vs 210) (Figure 3). Furthermore, while the user's subjective workload remained virtually the same, the system's estimated workload for that user, based on our tasking estimation model, showed that the overall estimated workload of the primary operator decreases when the system is used (Figure 4).

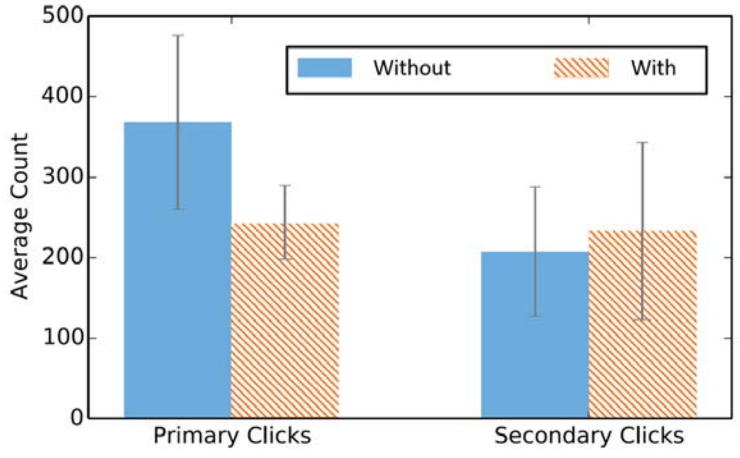

Figure 3. Average user interactions within the scenario.

So, while the specific workload numbers may or may not accurately reflect the user's subjective workload, because workload is estimated from the set of active tasks and events, this indicates that the primary user has fewer events high in his/her queue and that the offloading is actually functioning. As we see, the estimated workload when not using the system centers around 1.0, meaning that our workload calculations estimate that the primary user is almost always fully or nearly fully loaded. However, when using the system, the workload centers around about 0.6 indicating that the systems computes the operator's workload has reduced by about 0.4. This shows that events and tasks are being offloaded even if the effect is not easy to assess.

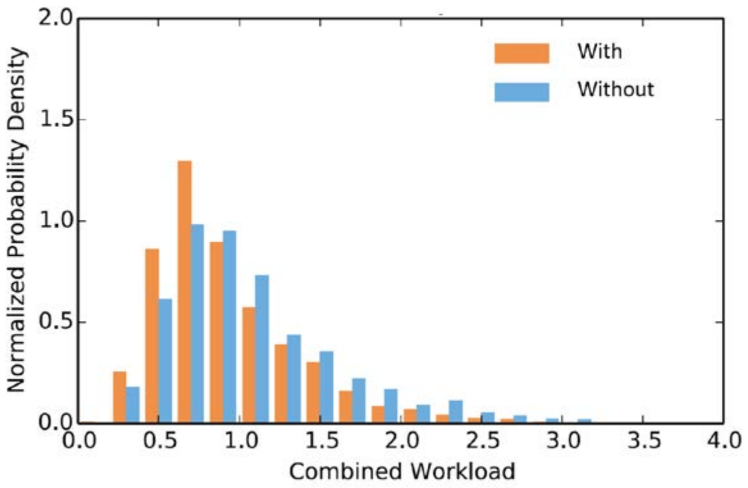

Figure 4. Distribution of Primary Operator Workload

While these results are encouraging, our workload system requires model tuning to be accurate/effective and during this evaluation, we did not have the time to go through multiple tuning/adjustment phases. Thus, our overall results reflect a first pass at modeling workload that we believe would improve significantly if further work were done, given the data collected. For example, the workload functions (*W*(t) in Equation 1) could be adjusted, at least for the manual component, based on the data we collected in the evaluation. Because we did not know how users would like to use the offloading capability, we structured the evaluation to allow the user significant freedom on how this was done. The users were allowed to offload their own events and assets if they wanted. Both users were allowed to task any assets they wanted to (leading, in some cases, to conflicts that had to be resolved via chat).

Furthermore, discussions after the evaluation led us to conclude that the offload method/algorithm should potentially be selectable by the user. That is, the system should implement multiple algorithms that the user can select among based on the mission and mission state.

Discussion and Future Directions

Our most important finding from our system evaluation is that the overall system appears to significantly reduce the manual interaction and intra-team interaction required for ISR missions using multiple UAVs. We hypothesize this effect is caused by (1) a reduction in the scan/check process required to maintain situation awareness and (2) better targeted tasking (few tasks issued, better sensor coverage). Though performance was improved and throughput was increased, subjectively, users did not report lower workload. This suggests that users were trading one type of work for another. In this case, we believe that the task allocation system reduced the user's manual workload allowing the primary user to focus more attention on cognitive and visual aspects of the problem. Our hypothesis is supported by the fact that operators using the enhancements were able to execute the most challenging parts of the mission, ground target tracking, more often and more effectively.

Despite the operational improvements, it was clear from our analysis that workload estimation and offloading requires additional refinement, especially to the workload model. This could be accomplished by using the evaluation data as a basis for model tuning/learning. For example, we could model each workload function as a probability distribution and learn the parameters of these distributions using user activity.

In addition, our current workload models are based only on *a priori* estimation of task workload and the combinatorial workload associated with multiple tasks. New learning models would be most effective if combined with physiological sensors that provided objective real-time workload measures such as heart rate variability (HRV) and pupillometry. For this approach, we would seek to leverage the considerable research in operational neuroscience (Schmorrow, Estabrooke, & Grootjen, 2009) and neuroergonomics (Parasuraman, 2003) which has shown the potential of real-time assessment of workload and situational awareness from behavioral and neurophysiological sensing. This physiological data is becoming surprisingly easy to measure. Several existing commodity fitness trackers are comfortable to wear for long durations while providing accurate measures of heart rate, galvanic skin response, respiration rate, and body temperature. In addition, other measures, such as eye tracking tools, are also becoming much more cost effective. This type of data holds promise to enhance real-time workload estimation by providing excellent data for offline training of workload models and real-time closed-loop feedback about user workload. This will improve the *a priori* task estimates over the long term and enable the estimation models to adapt to the specific user and operating conditions.

Acknowledgements

This research was sponsored by a Small Business Innovation Research (SBIR) award through NAVAIR PMA (contract #N68335-14-C-0015). The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied of the U.S. Navy or U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation herein.

References

Breslow, L., Gartenburg, D., McCurry, J.M., & Trafton, J.G. (2014). Dynamic operator overload: A model for predicting workload during supervisory control. *IEEE Transactions on Human-Machine Systems 44*(1), 30-40.

Parasuraman, R. (2003). Neuroergonomics: Research and Practice. *Theoretical Issues in Ergonomics Science 4*(1-2), 5-20.

Salas, E., Rosen, M., Burke, C.S., Nicholson, D., & Howse, W. (2007). Markers for enhancing team cognition in complex environments: The power of team performance diagnosis. *Aviation, Space, and Environmental Medicine 71*(1 supple), B77-B85.

Schmorrow, D.D., Estabrooke, I.V., & Grootjen, M. (2009). *Foundations of Augmented Cognition. Neuroergonomics and Operational Neuroscience 5th International Conference.* Berlin Heidelberg: Springer-Verlag.

Solovey, E., Zec, M., Garcia-Perez, E.A., Reimer, B., & Mehler, B. (2014). Classifying driver workload using physiological and driving performance data: Two field studies. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. Toronto, Canada: ACM.

Wickens, C.D. (2008). Multiple Resources and Mental Workload. *Human Factors 50*(3), 449-455.