Attention Guidance Agents with Eye-Tracking
A Use-Case Based on the MATBII Cockpit Task

Szonya Durant[1(✉)], Benedict Wilkins[2], Callum Woods[1], Emanuele Uliana[2], and Kostas Stathis[2]

[1] Department of Psychology Royal Holloway, University of London, Egham TW20 0EX, UK
szonya.durant@rhul.ac.uk, callum.woods.2014@live.rhul.ac.uk
[2] Department of Computer Science Royal Holloway, University of London, Egham TW20 0EX, UK
{benedict.wilkins.2014,emanuele.uliana.2016}@live.rhul.ac.uk, kostas.stathis@rhul.ac.uk

Abstract. A first step to keeping the human 'in the loop' in the context of developing intelligent multi-task interfaces is to be able to monitor their attention. By combining eye tracking with agent monitoring and decision making, we provide a basis for increasing the user's attentional bandwidth by offering bottom-up attention guidance. We develop a modified implementation of the MATBII cockpit task simulator embedded in an agent environment in which agents monitor events, including eye tracking, and act to deploy visual cues to guide attention. We explore how such a system may be useful for improving task performance, by also simulating users with agents to demonstrate how the system might work for some examples of user behaviour. We also discuss how our system can act as an experimental platform to benefit future user experience research focusing on attention guidance in complex multi-task interfaces.

Keywords: Agent environment · Eye-tracking · Interface · Workload

1 Introduction

Humans are often in complex, attention demanding situations, which require them to process information from multiple sources at once. In an interface such as an airplane cockpit many different information sources are present in the form of instrument displays spatially distributed in front of the pilot. Many other such examples exist from air traffic control to remote monitoring of autonomous vehicles in case of a required emergency intervention [16]. Humans are limited in their attentional capacity and thus sample parts of their environment sequentially over time [11]. When humans 'fail to notice' it is because of sub-optimal sampling. High information flow due to the number of displays, rapid information change in displays and the dependence of information between displays challenges human attention limits [43]. This may lead to poor decisions with serious consequences.

N. Alechina et al. (Eds.): EMAS 2021, LNAI 13190, pp. 92–113, 2022.
https://doi.org/10.1007/978-3-030-97457-2_6

Incorporating attention guidance with a complex multi-task interface is not straightforward, both from a conceptual and an implementation perspective. Existing approaches for building such systems, often focus on the conceptual aspects of attention guidance e.g. [15,45], but little attention has been paid to conceptual frameworks that also have a systematic implementation. Other approaches, e.g. [50], use agents as cognitive assistants, they perform autonomous situation assessment and take into account the limitations of human information processing. Still, an important aspect remains open: how can we build an agent system that considers how to convey information to users about ongoing operations and environmental parameters within their attentional limits?

The aim of this work is to show how to rethink attention guidance in multi-task interfaces using cognitive agents [7] that perceive where a user looks, and formulate interaction of display objects as events happening in an agent environment [8]. By observing the state of the various interface objects and in-coming user input data (including eye tracking data) and aggregating it to form beliefs, we conjecture that cognitive agents will be able to provide useful guidance to a user while making important considerations relating to their attention. The system thus both measures the current location of attention (based on eye tracking data) and alters attention by guiding the gaze tasks requiring input. Our specific objective is to exemplify the framework by developing the methods for attention guidance in the MATBII cockpit task simulator [59], showing how to organise guidance for a concrete application. We also wish to demonstrate how the modularity of an agent-based approach eases the process of experimentation and provides some unique benefits for creating a system that is extensible and reproducible.

The contribution of our work is to provide a practical system that makes use of gaze location (a proxy for spatial attention), allowing agents to use this information to help users allocate their limited cognitive resources. To this end we reproduce and improve some aspects of MATBII [59], producing our own simple simulation of a cockpit-based task space. The resulting system, which we refer to as ICU, allows for display changes and eye movements to be monitored externally via an event-based API, making it suitable for experimental settings beyond this work. We have also embedded ICU in a resource-light agent environment, which re-implements in full a single GOLEM container [8]. This supports real-time attention guidance mechanisms using cognitive agents to monitor where a user looks and can support attention guidance in other domains, assuming they provide an ICU like API.

The work is structured as follows. We begin by first outlining related work in describing and measuring attention, its limitations, cognitive workload, the use of agents as assistants, and assess to what extent MATBII has proved useful as an example task space. In Sect. 3 we present ICU, an open source Python implementation of the MATBII task space [59], which functions independently of our agent system. We then describe the agent system ICUa (ICU with agents) we have developed as an experimental platform for bottom-up attention guidance. In Sect. 4 we test the system by simulating and exploring some simple potential human behaviours. Finally in Sect. 5 we discuss the potential of our system,

the ability of agents to monitor the environment and user (via eye tracking) to provide useful attention guidance, and its suitability for future human testing and use in further applications.

2 Related Work

2.1 Workload, Eye-Tracking and Attention Guidance

The concept of workload and the demand on the limited attention of the human operator is important in human factors. Mental workload describes the demands on attention made by a cognitive task [42]. Often behavioural and physiological measures are used to try to classify situations as eliciting low or high mental workload e.g. [5,17,32,66]. A high mental workload has the effect of decreasing performance and increasing stress [42]. Often the aim of classifying high/low mental workload is to arrive at a solution aimed at alleviating conditions when high workload is detected, in the form of automation that can be introduced to aid the human operator. However, since the earliest introduction of automation, it has been suggested that in many situations it is important to keep the human in the loop even when a task has been devolved to an agent. The evidence suggests that at most levels of automation it is important that the human operator is kept engaged whenever possible user response might be required e.g. in the case of automation failure [21]. Thus even highly automated systems may need to consider how to convey information is such a way that the user is able to react - the basis of this work.

Multiple ongoing tasks lead to divided attention, which is particularly detrimental to performance [42]. There is a general trade-off between the need for selective attention to solve a given task and the need to detect other tasks that may require attention. In divided attention conditions with complex tasks, the phenomenon of cognitive tunnelling is often observed [41]. In this case if a user is focused on solving a particular task, even salient cues can be missed.

Warning lights and alerts are used in interfaces to capture the attention of the user, but this may lead to a situation where several alerts are activated at once leading to 'misplaced saliency' [6]. In this case the attempt to make an area stand out more has in fact the opposite effect by highlighting several areas and thus further overloading the human, as they have to decide which to attend to first. Additionally, overuse of alerting can lead to 'automation disuse' where the user comes to ignore the help that is being offered, seeing it as a nuisance [68]. These aspects of attention are key to understanding how to improve situation awareness (SA). SA describes a person's awareness of relevant aspects of their environment, the comprehension of these aspects, and predictions of what these will mean in the future [20]. Lacking situation awareness is one of the main causes of accidents attributed to human error [61].

In the context of describing how human attention is allocated over multiple displays, it is important to note that the spatial layout of these displays plays part in how the human user represents them [68]. Spatial memory is a key component in monitoring the work space, spatially reorganising parts of the

display has been found to be detrimental to performance [25]. Hence, it is generally preferable for an attention guidance system to maintain the layout of the interface to allow a spatial representation to form.

Eye tracking has proved an invaluable tool in attempting to measure workload, through indicators such as changes in pupil size or the duration of each fixation [40]. The spatial specificity of eye tracking has also led to it being developed as a tool for interaction [39]. A recent example uses eye tracking information to ascertain which screen the user is currently looking at in order to guide them to another screen in multi-monitor displays [63]. We propose that the spatial specificity of eye tracking could be used for more localised guidance.

2.2 Gaze Contingent Attention Guidance

There have been many proposals over the years on how to design 'attention aware systems' [54]. Concepts such as gaze based notifications have been introduced and evaluated according to their 'noticeability' vs 'distractiveness' [33]. A great deal of work has been done on gaze contingent attention guidance in the field of education and training where the learner's gaze is directed in an attempt to ensure optimal learning [56]. This is done by using online eye tracking to detect where the learner is focusing on the wrong information and using changes in the display to guide their attention - the same principles we intend to use in this work. A recent system for air traffic guidance makes use of online eye tracking to monitor the user's attention and direct it according to a simple logic that decides where the user should be looking [48]. This very specific implementation, with a control system tailored to air traffic control uses peripheral and central cues to guide attention to the necessary parts of the scene. Initial tests with five users suggested some improvement in perceived workload, although clear performance metrics relative to a baseline were not presented in this preliminary work. Earlier work [52] directs the user attention to target locations using a moving dot. In this work they do not consider rules for guiding attention, and the eye tracking and performance results are again not compared to a baseline. However, users reported positively on their interaction with the system, suggesting that this type of display has potential.

2.3 Agents

Human-computer environments where software agents act on behalf of a user are not a new idea e.g. [38], nor is automating tasks to reduce demands on human attention, e.g. [39]. Often agent capabilities have also been developed to predict intention or task state from behaviour i.e. overt responses and interactions, to provide assistance e.g. [50,57], and although eye-tracking agent assistants have been introduced, they still remain to be fully tested [65]. Adaptive interfaces have also been developed to use human physiological markers, such as heart rate and eye blinks to dynamically distribute tasks between agents and humans [28], but access to their corresponding test-beds is not available.

Cognitive assistants often use agent models to internalise perceptions as beliefs about the environment's state, actions to produce results (e.g. [37]) or use the BDI model (e.g. see [60]) based on intentions for goals the agent can plan for. Goal reasoning [2] allows goals to be achieved or maintained, including external goals specified by user guidelines and norms [58]. Agent decisions are modelled with preferences over planned goals using logic if there is certainty (e.g. [31]) or probabilities if there is uncertainty (e.g. [22]). Agent decisions may be explained (e.g. [44]) to build trust with the user - key to successfully working with a human [23]. However, many cognitive agent models and their implementation platforms (see [10,36]) are often resource heavy for real-time applications as demanding as eye-tracking. Although, light-weight versions exist, they are still at a prototypical stage [3]. In addition, the benefit of cognitive assistants for human performance has yet to be thoroughly evaluated experimentally in terms of assessing objective measures of performance compared to baseline - most evaluations rely on user questionnaire data reflecting subjective experiences [50].

To address some of the above limitations, our work is intended as a resource-light test-bed that combines agent environments and a teleo-reactive (TR) agent model [47] to support experiments for attention guidance applications where eye-tracking is a key requirement. TR agent models (e.g. [34]) and implementations (e.g. [13]) exist, and their link with models such as BDI have been studied (e.g. [14]). However, our work is the first to apply a resource-light TR model for attention guidance applications developed as agent environments.

2.4 MATBII as a Use-Case

MATBII [59] is widely used in the human factors literature as a multi-tasking space. It is comprised of clearly defined spatially separated sub-tasks often requiring rapid switching of attention. Difficulty is understood in terms of how often each sub-task needs attention, thus MATBII is often used to investigate low and high workload by changing the level of task difficulty, e.g. [24]. As shown in Fig. 1, the sub-tasks consist of a system monitoring task, checking for changes in colours of lights or positions of scales that require a mouse click response to return to correct state; a tracking task that requires keeping a target within a set of crosshairs; and a resource management task that requires manipulating pumps to keep fuel tanks at the right level. The pumps in the resource management task can be set to fail for a set amount amount of time. Pump failure is shown by a change in colour and the fuel level going out of range is also indicated by change in colour. MATBII is set up in such a way that under high frequency conditions the probability of 'misplaced salience' is high. A further important observation found from response patterns on MATBII is the presence of 'cognitive tunneling' as described above, manifesting itself as the inability to switch from one sub-task to another [24]. This provides us with multitasking situations, where it is objectively clear at any point what the user needs to look at.

There is not a great deal of literature on eye tracking users in MATBII [59]. Nelson et al. [46] report percentage time fixating on each task, Kim et al. [32] report changes in pupil size with increasing workload, and Berthelot et al. [5]

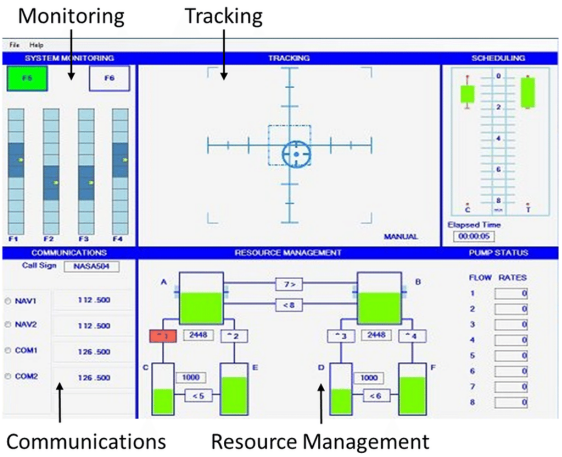


Fig. 1. The MATBII system with sub-tasks labelled. (Color figure online)

extract a property called 'self affinity' from eye movement statistics. There is much yet to be explored in the spatial pattern of eye movements whilst completing the task, for instance the effects of misplaced salience and cognitive tunnelling have only been inferred from behaviour, it would be useful to see these effects in more detail by measuring the spatial allocation of attention, which our proposed system allows for and at the same time uses this information to guide attention.

3 Integrated Cognitive User Assistance

3.1 ICU

Although an open source Python implementation of MATBII with eye-tracking (and further) options available has been recently released [12], we found it better suited to our purpose of combining the interface with an agent architecture to develop our own version of MATBII. We have opted for implementing a stripped down version of MATBII, essentially the same in functionality, using just a subset of the tasks but with some functional improvements that we feel are essential for experimentation. We call this system the Integrated Cognitive User (ICU[1]), which forms the interface part of the complete ICUa - with agents. Our system brings new scope for experiments in human factors research owing to more flexible manipulation of the task space, the ability to collect eye tracking data easily and interface in real time and also enables our work.

ICU has a bi-directional event API that may be used to interface with external programs and can be used in a number of ways, including monitoring the system in real time; for us its main purpose is to facilitate interaction with our agent

[1] https://dicelab-rhul.github.io/ICU/.

system. We have also tried to provide an improved configuration format[2], which can be used to quickly configure experiments by specifying event schedules concisely, and change aspects of the interface and task behaviours. Moreover, the system has built-in support for various kinds of user input, from standard input (e.g. keyboard/mouse) to eye tracking devices and could be easily extended to incorporate devices providing further physiological measures such as EEG or galvanic skin response. Devices are treated as part of the event system, device input is therefore exposed by the event API.

In terms of functionality, ICU reproduces the 'system monitoring', 'tracking', and 'resource management' tasks from MATBII [59] using Python 3, see Fig. 1. These tasks function similarly to those described in detail in [59]. Briefly, the system task involves responding to whether a green light switches off or a red light switches on, lights switch on/off according to a schedule, requiring a mouse click to reset to the correct state. It also includes a set of scales that change over time and that need to be kept as close the mid-point as possible and can be reset to mid-point by clicking on the scale level. The tracking task uses a joystick or keyboard presses to keep a randomly drifting target centred, the extent of the drift is configurable. The resource management task requires the user to switch pumps on and off to maintain the top two fuel tanks at the correct level, the pumps fail at certain times making them unusable, pump transfer rates, tank capacity, burn rate, frequency and duration of each pump failure, among other things can be configured.

To support eye-tracking, ICU provides a wrapper around the PsychoPy library [51], which enables any eye-tracker supported by the library to be used with ICU (we assume that the eye-tracker is already calibrated). The system was tested using a USB screen based X2-30 Tobii eye-tracker, sampling at 30 Hz on average. Raw gaze coordinates are filtered using an I-VT filter with standard moving average as specified in [49], coordinates are classed as fixation (eyes are stationary) or saccade (eyes are moving and thus unable to take in information).

3.2 ICUa: ICU with Agents

Previous work demonstrates the effectiveness of software agents for monitoring practical applications, e.g. see [9,35,55,67]. Here we extend these works conceptually, by introducing an agent environment that contains ICU as an internal object, where different agents can monitor the state of ICU (including information provided by an eye tracker) and perform actions on it to highlight parts of the screen for the user's benefit. Although our framework is demonstrated with ICU it is not specific to it, as ICU is used here more as an example to integrate any suitable multi-task interface.

An agent environments approach has some significant benefits from a software engineering perspective, especially modularity, which allows us to develop and swap out different objects and agent behaviours easily for experiments. Additionally, an agent-based approach leaves room for expanding the scope for

[2] https://dicelab-rhul.github.io/ICU/documentation/configuration/.

more complex environments by relying on multi-agent communication and coordination models, and as a way of integrating complex cognitive capabilities for guidance e.g. reinforcement learning [4].

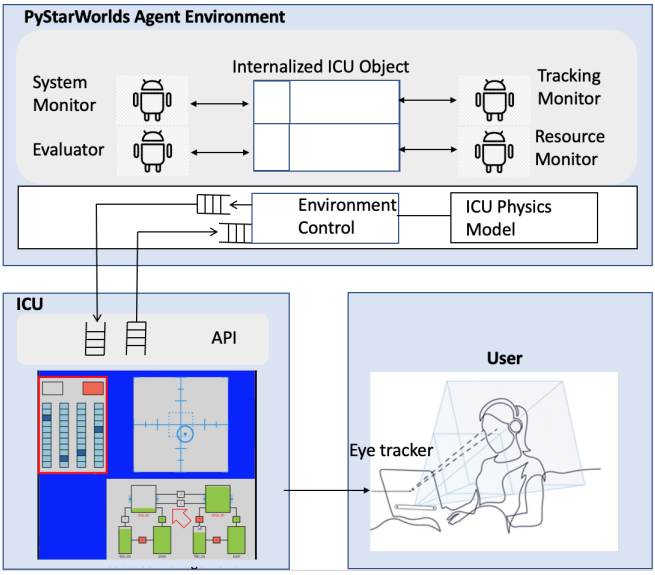


Fig. 2. ICUa reference architecture in PyStarWorlds showing the four agents deployed. We assign one agent to each of the first three application simulator tasks: system monitoring, resource monitoring and tracking. These agents subscribe to task specific events enabling them to perceive relevant information about the simulator's current state, including eye-tracking information about saccade or fixation, and communications from other agents in the system. The agents' actions have the effect of modifying the application interface i.e. to draw an overlay. We consider actions with two kinds of feedback (a) highlighting a particular sub-task and (b) draw an arrow at the current gaze location that points in the direction of a component that needs urgent attention. The fourth agent, the evaluator, monitors the user's performance using specific performance metrics.

ICUa is ICU extended with agents implemented in PyStarWorlds [1], an agent environment library that supports Python agent applications. The reference architecture of ICUa, shown in Fig. 2, is based on a specialised single container version of the GOLEM framework described in [7], which is implemented as an event-processing system under a publish/subscribe model [8]. ICU is internalised as an environment object by the API it exposes, so that its state can be perceived and acted upon by agents. Agents have a mind and body [62], the mind controls the agent behaviour, while the body relies on sensors and actuators to situate the agent in the application environment. Agents perceive events with their sensors, make decisions with their mind and attempt actions with their actuators. A

type-based publish/subscribe mechanism routes events to/from the sensors/actuators [8], which is known to be scalable. The environment provides a *Physics* module containing action execution rules where the semantics of each action are defined. We assume that agents are aware *a priori* of action preconditions/effects and so are able to decide which actions should be taken. ICUa is agnostic as to which agent model is to be used, different models can be adopted depending on the application domain.

For this domain, agents and their behaviours are specified in Python using condition-action rules following the teleo-reactive (TR) execution model [47] for goal directed behaviours (e.g. [18]). We assume a fixed *perceive-revise-decide-attempt* control cycle [30] that allows an agent to perceive the latest environment changes via the sensors, revise its internal state modelling the environment (or belief store), then decide about what action(s) to take, and finally attempt these actions using the agents actuators. In this setting, the TR model helps us structure the behaviours of the agent within the *decide* part of the agent's cycle, according to the goals the agent seeks to achieve. These behaviours are specified as a set of condition/action rules of the form:

$$G : \{C_1 \rightarrow A_1; C_2 \rightarrow A_2; \ldots; C_i \rightarrow A_i; \ldots; C_n \rightarrow A_n\}$$

where G is a goal, C_i a condition over internal variables (beliefs), and A_i is either a primitive action, or a sub-goal (giving rise to a sub-behaviour) that can itself be a TR program of the form:

$$A_i : \{C_{i,1} \rightarrow A_{i,1}; C_{i,2} \rightarrow A_{i,2}; \ldots; C_{i,m} \rightarrow A_{i,m}\}.$$

This gives rise to a significant simplification of a BDI-style planning layer that manipulates a plan library in which plans are comprised of hierarchical, suspendable and recoverable teleo-reactive programs [14]. The top-level goal G for the agent is triggered inside the *decide* part of the agent's cycle. The list of rules is scanned top-down for the first rule whose condition is satisfied, to select an intention and the corresponding action is attempted. It is important to note that the conditions are continuously being evaluated at each cycle step, so that when the first true condition changes due to new belief update, the intention changes accordingly. In other words, an action/sub-goal is revised, only when its true condition in the agent's internal state ceases to be true.

It is straightforward to create a subset of the TR paradigm for developing agent behaviours using Python, or a similar programming language. Assuming a round-robin agent execution of an agent's control cycle, there is a natural correspondence between TR programs and most programming languages, as shown in Fig. 3. An example of a simple monitoring behaviour that follows this model is given in Fig. 4(a). This kind of programming is quite flexible and can support more complex behaviours. For example, in principle an agent may be monitoring multiple parts of a screen (e.g. multiple pumps for the resource management task), it may attempt multiple actions in a single cycle (e.g. to highlight multiple pumps). As a result, the top-level goal in such cases needs to operate on sets of actions, simulating parallel execution of independent monitoring behaviours,

each with the form of a TR program, as for example in the interpretation used by [13], but in our case using PyStarWorlds. An example is given in Fig. 4(b).

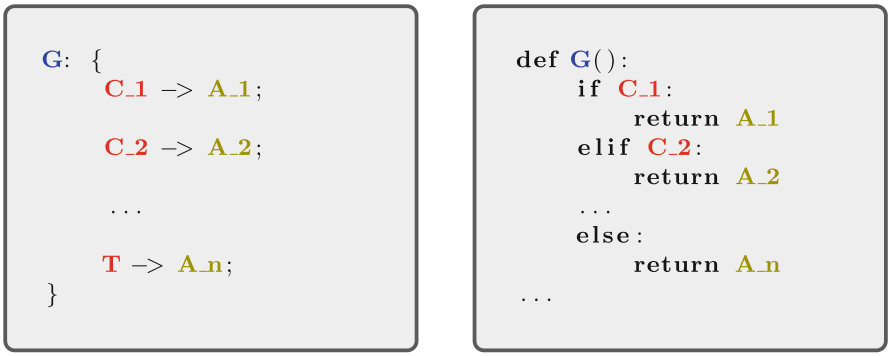


Fig. 3. Mapping of a simple version of TR rules in Python, which in PyStarWorlds are evaluated continuously. Sub-goals are method calls. `C_n = True` forces the last rule to always succeed if all other rules above fail to trigger.

Using the above architecture we implement a few simple rules for our agents to adhere to. The agents' goal is to shift the attention of the user to a sub-task that requires action if the user appears to have ignored it. Each agent has a built in grace period, which constitutes whether the sub-task is deemed to have been ignored. If in this time the required action has not taken place and importantly the gaze has not moved to the sub-task, then the agent responsible for the sub-task displays a relevant highlight. A highlight can be configured to constitute an outline of a sub-task, a transparent overlay, an arrow at the current fixation point or a combination of these. This involves agents checking the current gaze fixation position and whether it is in the required sub-task region of interest. Thus, we ensure that guidance is not displayed unnecessarily if attention has been transferred, but an action not yet produced. The agent also checks that no other guidance is being displayed at the time, as the aim is to not introduce a divided attention condition. So only one agent will be displaying guidance at any given time. If the requirements are met, the agent will display guidance and this will remain on display until the gaze position moves to the required sub-task or the task is resolved. Again, once gaze has moved we take this as an indicator that the task will be responded to as required. However, if the user moves their gaze away whilst the task still requires attention, it will again become highlighted after a second grace period, if the gaze has not returned. These simple rules are designed to move the user's attention on from a cognitive tunnelling situations with minimal unnecessary competing visual additions to the display. We do not assign differential importance to any of the sub-tasks, but such a hierarchy could easily be implemented in future.

```
def decide(self):
  # user is not looking -> highlight
  if not self.is_looking() and \
        not any(self.highlighted) and \
        not self.is_acceptable(self.component):
    return self.highlight(self.component)
  else:
    # default rule is T -> do nothing
    return None
```

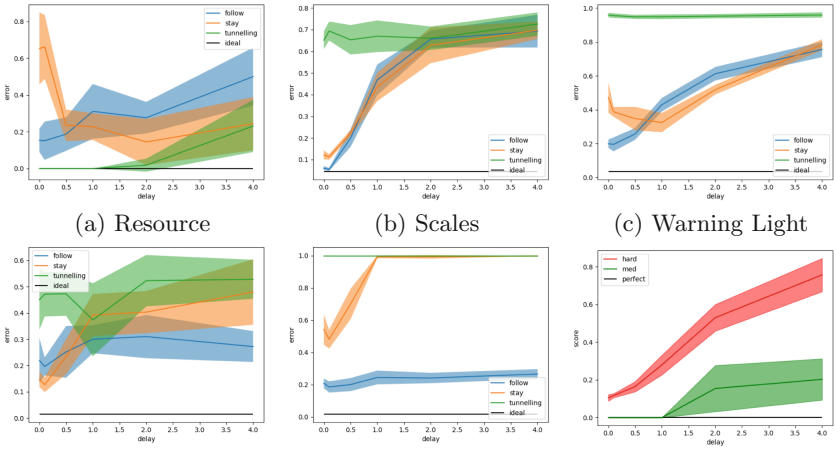
(a) Simple Monitoring Agent Example

```
def decide(self):
  actions = []

  # top level TR program
  if not any(self.highlighted):
    for component in self.task.components:
      # simple parallel TR program
      if not self.is_acceptable(component):
        action = self.highlight(component)
      else: # default rule is T -> None
        action = None

      actions.append(action)

  # default rule is implicit T -> {}
  return actions
```

(b) Parallel Monitoring Agent Example

Fig. 4. Agent monitoring examples in Python TR style. In (a) we show a simple single-action monitoring behaviour that highlights a component (part of a task) if needed. In (b), we operate over sets of actions. This enables the agent to highlight many components if necessary. In practice we limit agents to highlighting a single component (to avoid overloading the user), however the parallel execution of behaviours is useful for our simulated human users outlined in Sect. 4.

4 Simulating Simple Examples of Human Behaviour



(a) Resource (b) Scales (c) Warning Light

(d) Tracking (e) Highlight (f) Workload

Fig. 5. Graphs (a)–(e) show error of three simulated users according to the evaluation metrics given in Sect. 4.1, lower error means better performance by the agent. The 'stay' user will ignore agents advice until a particular task is complete (there is not more action to be taken for the moment). The 'follow' user will always take the agents advice and move to solve the recommended task (see Fig. 6 for details). Error is shown as a function of the "delay" period introduced to each decision, a restriction on the user's ability. There is no delay parameter for the ideal (ideal) user, we mark this minimal possible error as a horizontal line on the graphs for comparison. All results are averaged over 10 runs of 1 min each and normalised in the [0–1] interval. The shaded regions show 95% confidence intervals. Graph (f) shows the error for the 'tunnelling' user at two difficulty levels (medium and hard) on the resource management task. The difficulty is set in the configuration file by specifying different event frequencies. For all simulations grace periods until a highlight is displayed are set to 2 s.

To demonstrate the flexibility of our agent system and provide some insight into how the system might perform with different user behaviours we have implemented and evaluated four different kinds of 'user' agents, see Fig. 5. Each 'user' agent directly observes events from the ICU system and is able to provide *fake* user input e.g. mouse clicks, key input and eye movement. This set up also provides a basis for future researchers wishing to simulate more complex human behaviour.

The ideal user reacts immediately to MATBII events and is not constrained by any input delay e.g. eye movement speed or response delay, it can simultaneously observe and react to changes in all tasks. This ideal agent is used as a baseline and achieves the highest possible performance (i.e. lowest error rate with no need for guidance).

We also model a worst case scenario with a user agent that never moves their eyes from a sub-task (which in this case is the resource management task) and only makes responses to that sub-task, a case of cognitive tunnelling ('tunnelling').

The final two users are imperfect, in that they only respond to a sub-task once guidance is provided and they require some time to provide a response. They will only attempt to solve a sub-task when looking (i.e. while fixated on the sub-task area) and require time to act (including eye movement). In our experiments an agent moves its eyes at a constant speed 1000px/s mimicking the rapid saccades made my humans in between fixations, which we model here as the gaze remaining static in a given location. The relevant part of the two behaviours is presented in Fig. 6.

```
if self.is_highlighted(task):
  return self.move_eye(task)
elif self.is_looking(task) and not self.solved(task):
  return self.solve(task)
```

(a) Follow

```
if self.is_looking(task) and not self.solved(task):
  return self.solve(task)
elif self.is_highlighted(task):
  return self.move_eye(task)
```

(b) Stay

Fig. 6. Two exerts from the simulated imperfect users showing the key difference in their behaviour. (a) The 'follow' user's gaze always follows guidance when present causing it to abandon its current, possibly unresolved task and move to another. (b) the 'stay' user remains focused on a task until no further action can be taken to resolve it then follows guidance to reach the next task.

The two follow/stay behaviours are set up to correspond to two extremes of behaviour, we expect human behaviour to lie somewhere in-between the two. With both, we vary the delay with which they are able to respond. With larger delay times, it is more likely that the 'follow' user will abandon a sub-task before it has been solved, while with the 'stay' user, other tasks will remain unsolved for a longer period.

4.1 Evaluation Metrics

To measure the performance of each user agent we use the following metrics, which are normalised over time and averaged over components where applicable. The metrics are representative of the user's error when solving tasks, 0 being a perfect score for each metric.

- Time that main fuel tanks are out of the acceptable range in the resource management task.
- Deviation from acceptable state of the scales ($\mathcal{L}_1/time$) in the system monitoring task.
- Time that warning lights are in an incorrect state in the system monitoring task.
- Deviation from the central acceptable box ($\mathcal{L}_\infty/time$) in the tracking task.
- Time at least one warning (highlight or otherwise) is displayed on the overlay.

4.2 Simulation Results

The errors calculated using our evaluation metrics are shown in Fig. 5. The ideal user (see Fig. 5) has minimal error, any small error that exists is a result of the slight processing delay due to simulation speed (100 ms per agent cycle). If we look at the performance measure associated with the length of time that highlights are displayed, we see that this is zero for the ideal observer, reflecting that our system does not display highlights when not required.

At the other extreme our 'worst case' tunnelling user (Fig. 5) provides an upper bound level of error on tasks other than the resource management task they are focusing on. The on/off nature of the warning lights is reflected in the constant maximum error across delay, the scales and tracking are more variable in their error as it is possible for them to randomly return within acceptable parameters. For this user, after the first grace period, a highlight will always be displayed. In the resource management task of course they perform best as this is the task of focus. We can see the effect of delay in the responses making their performance worse.

In the case of our imperfect users that are guided by the highlights (the follow and stay users in Fig. 5), but do not respond otherwise, their performance is somewhere between the two more extreme ideal and tunneling users, as expected. This reflects that a human who makes use of the highlights to guide them, is of course not a perfect user, but is likely to perform better than a user who completely ignores the need for a response. With these users we can also see the effects of a delay in the response, as the response slows, so we can see the error generally increases. The advantage of staying on a task ('stay' user Fig. 5) until it is 'solved' varies to some extent with the delay of the response. The warning lights and scales parts of the system monitoring task suggest an initial small advantage for always following the highlighting, which then disappears with delay. The more continuous nature of the tracking task produces a different pattern, with an apparent small initial advantage for the user who remains on task until solved,

but with increased delay the user that follows the highlighting performs better. The highlighting metric reflects these two behaviours in that the user that follows the highlighting has less highlights on screen over time. Of course we also see that following highlights will reduce performance relative to the tunnelling approach on the single task chosen to focus on (resource management).

In Fig. 5 (f) we illustrate how task difficulty can be manipulated in our system by changing the frequency of events. We show the results for our worst case 'tunnelling' user on the resource management task. With a short delay in the user response there is a relatively small difference in performance between low frequency and high frequency events, as they are able to respond quickly enough to resolve the high frequency events. With increased response delay, in each case the user performs worse, with a higher error for the more difficult case consistently.

Our simulations have shown how attention guidance may in principle improve performance for imperfect users in cases where users shift their attention immediately, compared to when they are unable to shift their attention due to cognitive tunnelling. Our guidance agents' behaviour has been tailored in an attempt to provide the most useful feedback and avoid overloading the user. We have tested only one class of guidance behaviour, based on the principles outlined in Sect. 2, which works as a proof of concept. Our simulations also allowed us to visualise the effect of increasing task difficulty by increasing event frequency and how this depends on the delay in the user action.

In addition to our simulated user tests, we have tested the capacity of the system and found that the ICUa was able to deal with up to a million events per second without raising any performance issues (for reference, the event load under normal operation does not exceed more than one thousand per second with a high-throughput eye tracking device).

5 Discussion

We have successfully demonstrated, by modifying the widely used MATBII cockpit task simulator, how an information display and interface system can be monitored by agents and how a human user may be incorporated into the agent environment by monitoring of their eye movements and responses. Our agents have been deployed to enact simple attention guidance in a simulated setting. We have demonstrated this important test case as a proof of concept of the architecture of such a system in a simple task space known to replicate some of the problems that have been found with user inattention.

5.1 Simulation Summary

We used our agent system to build 'user agents' that are able to simulate some simplified examples of human behaviour synthetically. This enabled us to demonstrate the behaviour of the system by summarizing error patterns under different

conditions. We can conclude that the system works as expected, and that following the guidance reduces the error from a worst case scenario where a user is only paying attention to a single task. The different simulated users showed different patterns across delay. Changing the rules we implemented for following the highlights resulted in different performance error patterns. We also demonstrated how the configuration can be set to manipulate the difficulty of a task with a resulting change in performance.

The user agents are designed to demonstrate only upper and lower bound performance, and the effect of following the attention guidance for improving performance. We expect human behaviour to be some combination of our simulated users. Under certain conditions humans will be able to respond with some delay to a sub-task that required a response; sometimes they will only respond when there is a highlight; and sometimes the highlight may cause them to move their attention before they have solved a sub-task. This system is now suitable for experiments with human users to explore these scenarios and to ascertain optimal rules for the agents. For instance, the simulations suggest that highlights may not always be advantageous if the user follows them before they have solved the current sub-task they are focusing on.

The inclusion of user agents opens up our system to further simulations using more complex examples of human behaviours that may occur under different conditions and to test how ideal display rules may vary with different examples of human behaviour.

5.2 Future Experiments with Human Users

ICUa runs on a desktop PC with an eye tracker attached and can record the performance of human participants under different specified conditions. A first step would be to test the current system with the existing simple rules and assumptions to determine if it is effective at guiding attention and thus improving performance in humans. From the simulations it is already evident that there will be certain conditions under which attention guidance is particularly useful. In a low workload condition it may be that user guidance has lesser impact as there are less demands on attention, although studies also show negative effects of low expectancy - very infrequent unexpected events can also be missed, especially if there are other tasks that require constant monitoring [68]. If the events are happening too quickly for human users to successfully deal with them their may be a floor effect where attention guidance no longer helps (as seen in the longer delay times in our simulations).

Experiments would involve manipulating the frequency of events in our system and also comparing highlighting alone vs arrows alone and the two presented together, to examine the cost-benefit of single vs multiple and central vs peripheral cues. We expect to find a 'sweet spot' where attention guidance works best. The system can be combined with subjective measures of workload such as the NASA-TLX [26] as used within the original MATBII [59].

The modified ICU interface makes its suitable for measuring eye movements during a task that is similar to MATBII, making it suitable for wider experimen-

tation beyond our current set-up. This environment provides an ideal testing ground, for different methods of attracting and maintaining attention. Attention guidance could be varied by choosing different colours of highlights, or implementing synchronous flashing between the highlight and the arrow [68] or by blurring areas that are not of interest [27]. The current system allows for measuring associated eye movements, responses and performance with such changes. Moreover, by making use of the agent architecture, our simple rules for deploying attention guidance could be altered to observe the best effects on human performance. The use of agents provides a useful way of manipulating rules for changing displays.

5.3 Potential Further Extensions and Applications

As highlighted, one strength of the agent-oriented approach and our agent model is that it is modular - extra modules can be added in terms of additional interface tasks and associated inputs, but also in terms of physiological signals of attention and other measures of the human mental state we may want to represent. Not just visual, but also auditory inputs for example are possible and additional physiological measures can take us beyond tracking spatial visual attention, including other measures that can be read from the eye tracker such as pupil size. Pupil size has been a useful measure in terms of tracking vigilance, fatigue and workload [53].

As more complex inputs are added, so the agent behaviour repertoire can be expanded. More complex rules can be added, leading to the agents performing more complex calculations that exceed human capacity, defining for example what would be the best thing to attend to for the human, when this is no longer intuitively clear, especially under a moment of high pressure, or taking over some of the task and carrying out some of the required responses automatically. This could be done in an adaptive way [29], incorporating workload in to the agent model to enable adaptive processes, making the most of agents' human-like ability and explainability. The aim of the explainability is to help the human interpret the environment and the actions of the agents. Using agent behaviour that can be transparent to the human helps build trust, which is critical to optimal human computer interaction [19]. There is no explicit user modelling in the ICUa currently, however agents are particularly suited to more complex user modelling such as those used to track learning through tutoring software [67] and our system is suited for this kind of extension.

Agents can also provide the basis for a learning framework. Whilst agent behaviour may in itself alter due to the ongoing conditions, such as ongoing high workload or fatigue as a way of achieving goals under different conditions, a further degree of individualisation to the user may be possible by enabling agents to learn form the past behaviour of the user.

Mobile eye trackers can be used to map eye position in real time to the surrounding environment recorded by a camera [64]. It has been suggested that gaze based interactive displays could be useful in a cockpit setting [39], which MATBII is set up to mimic some aspects of. Current AI cockpit applications

involve automating many systems, which involves the human user handing over control. A future application of our system may be providing a way to ensure that human monitoring remains interactive, to keep the human in the loop in a cockpit environment. Increasingly cockpits are augmented, often being displayed in helmet in heads-up displays that can be moved around and tailored to the user, something that could be incorporated in to the agent system. The system we have developed aims to ensure that once a target for attention is known to the system, it is successfully processed by the human user. This emphasis means our work has applications in many systems where attention guidance might be called for, such as in semi-autonomous vehicles, within the remote operator room for automated vehicle systems, in air traffic control, or even alerts that may go unnoticed or not fully comprehended in everyday office computer usage.

6 Conclusion

We have built an attention guidance system using agent environments as the underlying framework. Central to our work is the notion that an interactive computer system construed as an agent environment should represent the human user as an entity providing continuing feedback, so that the system can ensure that they can process information within their limited attentional resources in order to produce the necessary human responses. Our proof of concept prototype aims to keep the human in the loop, in this case primarily via their eye movements and with feedback from agents. Our agent-based approach to attention guidance presents some clear advantages, such as modularity, scalability and extensibility. We propose that our approach, as exemplified by our system, is suitable for a wide range of experimentation where humans interact with multi-display interfaces based on attention guidance, and this is our next step for continuing this research.

Acknowledgements. The work for developing this demonstrator was supported by a *Human-Like Computing* EPSRC Network+ Kickstart grant. Thanks to Suzy Broadbent and Lisa Boyce for discussions and support.

References

1. PyStarWorlds (2021). https://github.com/dicelab-rhul/pystarworlds
2. Aha, D.W.: Goal reasoning: foundations, emerging applications, and prospects. AI Mag. **39**(2), 3–24 (2018)
3. Aschermann, M., Kraus, P., Müller, J.P.: LightJason. In: Criado Pacheco, N., Carrascosa, C., Osman, N., Julián Inglada, V. (eds.) EUMAS/AT -2016. LNCS (LNAI), vol. 10207, pp. 58–66. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-59294-7_6
4. Bagga, P., Paoletti, N., Alrayes, B., Stathis, K.: A deep reinforcement learning approach to concurrent bilateral negotiation. In: Proceedings of the 29th International Joint Conference on Artificial Intelligence, IJCAI 2020, pp. 297–303. IJCAI/AAAI, Yokohama (2020)

5. Berthelot, B., Mazoyer, P., Egea, S., André, J.M., Grivel, É., Legrand, P.: Self-affinity of an aircraft pilot's gaze direction as a marker of visual tunneling. Technical report, SAE Technical Paper (2019). https://doi.org/10.4271/2019-01-1852
6. Bolstad, C., Costello, A., Endsley, M.: Bad situation awareness designs: what went wrong and why. In: Proceedings of the 16th World Congress of International Ergonomics Association (2006)
7. Bromuri, S., Stathis, K.: Situating cognitive agents in GOLEM. In: Weyns, D., Brueckner, S.A., Demazeau, Y. (eds.) EEMMAS 2007. LNCS (LNAI), vol. 5049, pp. 115–134. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-85029-8_9
8. Bromuri, S., Stathis, K.: Distributed agent environments in the ambient event calculus. In: Gokhale, A.S., Schmidt, D.C. (eds.) Proceedings of the Third ACM International Conference on Distributed Event-Based Systems, DEBS 2009, Nashville, Tennessee, USA, 6–9 July 2009. ACM (2009)
9. Bunch, L., et al.: Software agents for process monitoring and notification. In: Proceedings of the 2004 ACM Symposium on Applied Computing, SAC 2004, pp. 94–100. Association for Computing Machinery, New York (2004). https://doi.org/10.1145/967900.967921
10. Calegari, R., Ciatto, G., Mascardi, V., Omicini, A.: Logic-based technologies for multi-agent systems: a systematic literature review. Auton. Agent. Multi-agent Syst. **35**(1), 1–67 (2020). https://doi.org/10.1007/s10458-020-09478-3
11. Carrasco, M.: Visual attention: the past 25 years. Vis. Res. **51**(13), 1484–1525 (2011). https://doi.org/10.1016/j.visres.2011.04.012
12. Cegarra, J., Valéry, B., Avril, E., Calmettes, C., Navarro, J.: OpenMATB: a multi-attribute task battery promoting task customization, software extensibility and experiment replicability. Behav. Res. Methods **52**(5), 1980–1990 (2020). https://doi.org/10.3758/s13428-020-01364-w
13. Clark, K., et al.: A framework for integrating symbolic and sub-symbolic representations. In: Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, pp. 2486–2492. AAAI Press (2016)
14. Coffey., S., Clark., K.: A hybrid, teleo-reactive architecture for robot control. In: Proceedings of the 2nd International Workshop on Multi-agent Robotic Systems - MARS, (ICINCO 2006), pp. 54–65. INSTICC, SciTePress (2006). https://doi.org/10.5220/0001225300540065
15. Danieau, F., Guillo, A., Doré, R.: Attention guidance for immersive video content in head-mounted displays. In: 2017 IEEE Virtual Reality (VR), pp. 205–206 (2017). https://doi.org/10.1109/VR.2017.7892248
16. Davies, A.: The war to remotely control self-driving cars heats up. Wired. https://www.wired.com/story/designated-driverteleoperations-self-driving-cars/. Accessed 11 Nov 2019
17. Debie, E., et al.: Multimodal fusion for objective assessment of cognitive workload: a review. IEEE Trans. Cybern. (2019). https://doi.org/10.1109/TCYB.2019.2939399
18. Dongol, B., Hayes, I.J., Robinson, P.J.: Reasoning about goal-directed real-time teleo-reactive programs. Formal Aspects Comput. **26**(3), 563–589 (2014)
19. Dzindolet, M.T., Peterson, S.A., Pomranky, R.A., Pierce, L.G., Beck, H.P.: The role of trust in automation reliance. Int. J. Hum. Comput. Stud. **58**(6), 697–718 (2003). https://doi.org/10.1016/S1071-5819(03)00038-7
20. Endsley, M.R.: Designing for Situation Awareness: An Approach to User-Centered Design. CRC Press, Boca Raton (2016)

21. Endsley, M.R., Kiris, E.O.: The out-of-the-loop performance problem and level of control in automation. Hum. Factors **37**(2), 381–394 (1995). https://doi.org/10.1518/001872095779064555
22. Fern, A., Natarajan, S., Judah, K., Tadepalli, P.: A decision-theoretic model of assistance. J. Artif. Intell. Res. **50**, 71–104 (2014). https://doi.org/10.1613/jair.4213
23. Glass, A., McGuinness, D.L., Wolverton, M.: Toward establishing trust in adaptive agents. In: Proceedings of the 13th International Conference on Intelligent User Interfaces, pp. 227–236 (2008). https://doi.org/10.1145/1378773.1378804
24. Gutzwiller, R.S., Wickens, C.D., Clegg, B.A.: Workload overload modeling: an experiment with MATBII to inform a computational model of task management. In: Proceedings of the Human Factors and Ergonomics Society Annual Meeting, vol. 58, pp. 849–853. SAGE Publications, Los Angeles (2014). https://doi.org/10.1177/1541931214581179
25. Hancock, P., Scallen, S.: The performance and workload effects of task re-location during automation. Displays **17**(2), 61–68 (1997). https://doi.org/10.1016/S0141-9382(96)01018-9
26. Hart, S.G., Staveland, L.E.: Development of NASA-TLX (task load index): results of empirical and theoretical research. In: Advances in Psychology, vol. 52, pp. 139–183. Elsevier (1988). https://doi.org/10.1016/S0166-4115(08)62386-9
27. Hata, H., Koike, H., Sato, Y.: Visual guidance with unnoticed blur effect. In: Proceedings of the International Working Conference on Advanced Visual Interfaces, pp. 28–35 (2016). https://doi.org/10.1145/2909132.2909254
28. Hou, M., Kobierski, R.D., Brown, M.: Intelligent adaptive interfaces for the control of multiple UAVs. J. Cogn. Eng. Decis. Mak. **1**(3), 327–362 (2007). https://doi.org/10.1518/155534307X255654
29. Inagaki, T., et al.: Adaptive automation: sharing and trading of control. In: Handbook of Cognitive Task Design, vol. 8, pp. 147–169 (2003)
30. Kakas, A., Mancarella, P., Sadri, F., Stathis, K., Toni, F.: Declarative agent control. In: Leite, J., Torroni, P. (eds.) CLIMA 2004. LNCS (LNAI), vol. 3487, pp. 96–110. Springer, Heidelberg (2005). https://doi.org/10.1007/11533092_6
31. Kakas, A., Mancarella, P., Sadri, F., Stathis, K., Toni, F.: Computational logic foundations of KGP agents. J. Artif. Intell. Res. **33**, 285–348 (2008). https://doi.org/10.1613/jair.2596
32. Kim, J.H., Yang, X.: Applying fractal analysis to pupil dilation for measuring complexity in a process monitoring task. Appl. Ergon. **65**, 61–69 (2017). https://doi.org/10.1016/j.apergo.2017.06.002
33. Klauck, M., Sugano, Y., Bulling, A.: Noticeable or distractive? A design space for gaze-contingent user interface notifications. In: Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems, pp. 1779–1786 (2017). https://doi.org/10.1145/3027063.3053085
34. Kowalski, R.A., Sadri, F.: Teleo-reactive abductive logic programs. In: Artikis, A., Craven, R., Kesim Çiçekli, N., Sadighi, B., Stathis, K. (eds.) Logic Programs, Norms and Action. LNCS (LNAI), vol. 7360, pp. 12–32. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-29414-3_3
35. Krauth, E.I., van Hillegersberg, J., Van De Velde, S.L.: Agent-based human-computer-interaction for real-time monitoring systems in the trucking industry. In: 2007 40th Annual Hawaii International Conference on System Sciences (HICSS 2007), p. 27 (2007). https://doi.org/10.1109/HICSS.2007.52
36. Kravari, K., Bassiliades, N.: A survey of agent platforms. J. Artif. Soc. Soc. Simul. **18**(1) (2015). https://doi.org/10.18564/jasss.2661

37. Laird, J.E.: The SOAR Cognitive Architecture. MIT Press, Cambridge (2012)
38. Maes, P.: Agents that reduce work and information overload. In: Readings in Human-Computer Interaction, pp. 811–821. Elsevier (1995). https://doi.org/10.1016/B978-0-08-051574-8.50084-4
39. Majaranta, P., Bulling, A.: Eye tracking and eye-based human–computer interaction. In: Fairclough, S.H., Gilleade, K. (eds.) Advances in Physiological Computing. HIS, pp. 39–65. Springer, London (2014). https://doi.org/10.1007/978-1-4471-6392-3_3
40. Marshall, S.P.: The index of cognitive activity: measuring cognitive workload. In: Proceedings of the IEEE 7th Conference on Human Factors and Power Plants, p. 7. IEEE (2002). https://doi.org/10.1109/HFPP.2002.1042860
41. Martens, M., Van Winsum, W.: Measuring Distraction: The Peripheral Detection Task. TNO Human Factors, Soesterberg (2000)
42. Matthews, G., Davies, D.R., Stammers, R.B., Westerman, S.J.: Human Performance: Cognition, Stress, and Individual Differences. Psychology Press, Hove (2000)
43. Matthews, T., Dey, A.K., Mankoff, J., Carter, S., Rattenbury, T.: A toolkit for managing user attention in peripheral displays. In: Proceedings of the 17th Annual ACM Symposium on User Interface Software and Technology, pp. 247–256 (2004). https://doi.org/10.1145/1029632.1029676
44. Miller, T.: Explanation in artificial intelligence: insights from the social sciences. Artif. Intell. **267**, 1–38 (2019). https://doi.org/10.1016/j.artint.2018.07.007
45. Navalpakkam, V., Itti, L.: A goal oriented attention guidance model. In: Bülthoff, H.H., Wallraven, C., Lee, S.-W., Poggio, T.A. (eds.) BMCV 2002. LNCS, vol. 2525, pp. 453–461. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-36181-2_45
46. Nelson, J.M., Phillips, C.A., McKinley, R.A., McIntire, L.K., Goodyear, C., Monforton, L.: The effects of transcranial direct current stimulation (tDCS) on multitasking performance and oculometrics. Mil. Psychol. **31**(3), 212–226 (2019). https://doi.org/10.3389/fnhum.2016.00589
47. Nilsson, N.J.: Teleo-reactive programs for agent control. J. Artif. Int. Res. **1**(1), 139–158 (1994). https://doi.org/10.1613/jair.30
48. Ohneiser, O., Gürlük, H., Jauer, M.L., Szöllősi, Á., Balló, D.: Please have a look here: successful guidance of air traffic controller's attention (2019)
49. Olsen, A.: The Tobii IVT Fixation Filter Algorithm description (2012)
50. Onken, R., Walsdorf, A.: Assistant systems for aircraft guidance: cognitive man-machine cooperation. Aerosp. Sci. Technol. **5**(8), 511–520 (2001). https://doi.org/10.1016/S1270-9638(01)01137-3
51. Peirce, J., MacAskill, M.: Building Experiments in PsychoPy. Sage, Thousand Oaks (2018)
52. Poitschke, T., Laquai, F., Rigoll, G.: Guiding a driver's visual attention using graphical and auditory animations. In: Harris, D. (ed.) EPCE 2009. LNCS (LNAI), vol. 5639, pp. 424–433. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-02728-4_45
53. Pomplun, M., Sunkara, S.: Pupil dilation as an indicator of cognitive workload in human-computer interaction. In: Proceedings of the International Conference on HCI, vol. 273 (2003)
54. Roda, C., Thomas, J.: Attention aware systems: theories, applications, and research agenda. Comput. Hum. Behav. **22**(4), 557–587 (2006). https://doi.org/10.1016/j.chb.2005.12.005

55. Rodden, T.A., Fischer, J.E., Pantidi, N., Bachour, K., Moran, S.: At home with agents: exploring attitudes towards future smart energy infrastructures. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI 2013, pp. 1173–1182. Association for Computing Machinery, New York (2013). https://doi.org/10.1145/2470654.2466152
56. Rosch, J.L., Vogel-Walcutt, J.J.: A review of eye-tracking applications as tools for training. Cogn. Technol. Work **15**(3), 313–327 (2013). https://doi.org/10.1007/s10111-012-0234-7
57. Sadri, F.: Logic-based approaches to intention recognition. In: Handbook of Research on Ambient Intelligence and Smart Environments: Trends and Perspectives, pp. 346–375. IGI Global (2011). https://doi.org/10.4018/978-1-61692-857-5.ch007
58. Sadri, F., Stathis, K., Toni, F.: Normative KGP agents. Comput. Math. Organ. Theory **12**(2–3), 101–126 (2006)
59. Santiago-Espada, Y., Myer, R.R., Latorella, K.A., Comstock, J.R., Jr.: The multi-attribute task battery II. A user's guide (MATB-II) software for human performance and workload research (2011)
60. de Silva, L., Meneguzzi, F., Logan, B.: BDI agent architectures: a survey. In: Bessiere, C. (ed.) Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020, pp. 4914–4921. ijcai.org (2020). https://doi.org/10.24963/ijcai.2020/684
61. Stanton, N.A., Chambers, P.R., Piggott, J.: Situational awareness and safety. Saf. Sci. **39**(3), 189–204 (2001). https://doi.org/10.1016/S0925-7535(01)00010-8
62. Stathis, K., Kakas, A.C., Lu, W., Demetriou, N., Endriss, U., Bracciali, A.: PROSOCS: a platform for programming software agents in computational logic. In: Müller, J., Petta, P. (eds.) Proceedings of the Fourth International Symposium "From Agent Theory to Agent Implementation" (AT2AI-4 - EMCSR 2004 Session M), Vienna, Austria, pp. 523–528 (2004)
63. Stratmann, T.C., Kempa, F., Boll, S.: Lame: light-controlled attention guidance for multi-monitor environments. In: Proceedings of the 8th ACM International Symposium on Pervasive Displays, pp. 1–5 (2019). https://doi.org/10.1145/3321335.3324935
64. Tatler, B.W., Hansen, D.W., Pelz, J.B.: Eye movement recordings in natural settings. In: Klein, C., Ettinger, U. (eds.) Eye Movement Research. SNPBE, pp. 549–592. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-20085-5_13
65. Toreini, P., Morana, S.: Designing attention-aware business intelligence and analytics dashboards. In: Designing the Digital Transformation: DESRIST 2017 Research in Progress Proceedings of the 12th International Conference on Design Science Research in Information Systems and Technology, Karlsruhe, Germany. 30 May–1 June, pp. 64–72. Karlsruher Institut für Technologie (KIT) (2017). https://doi.org/10.5445/IR/1000069452
66. Van Orden, K.F., Limbert, W., Makeig, S., Jung, T.P.: Eye activity correlates of workload during a visuospatial memory task. Hum. Factors **43**(1), 111–121 (2001). https://doi.org/10.1518/001872001775992570
67. Wang, H., Chignell, M., Ishizuka, M.: Empathic tutoring software agents using real-time eye tracking. In: Proceedings of the 2006 Symposium on Eye Tracking Research & Applications, ETRA 2006, pp. 73–78. Association for Computing Machinery, New York (2006). https://doi.org/10.1145/1117309.1117346
68. Wickens, C.D., Hollands, J.G., Banbury, S., Parasuraman, R.: Engineering Psychology and Human Performance. Psychology Press, Hove (2015)