

Context-Aware Online Adaptation of Mixed Reality Interfaces

David Lindlbauer, Anna Maria Feit, Otmar Hilliges
ETH Zurich, Switzerland

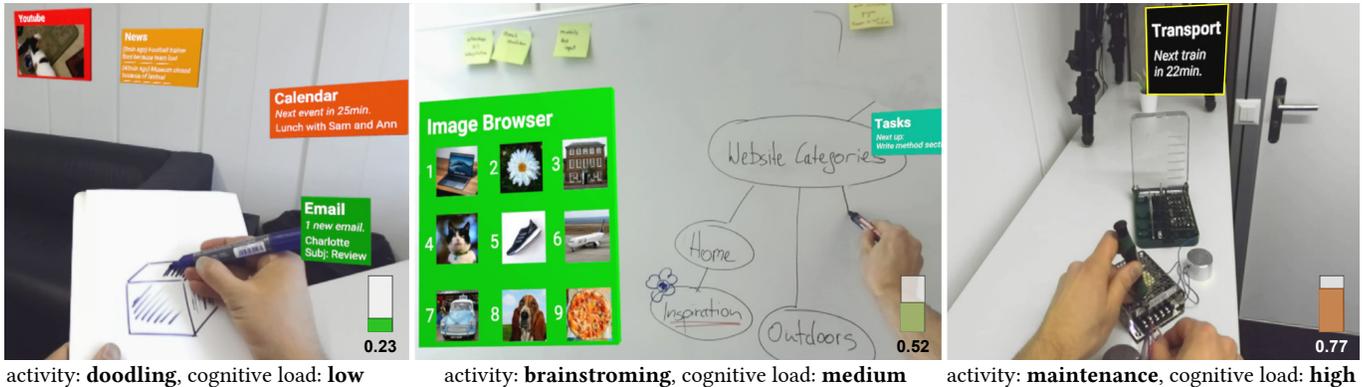


Figure 1. We propose an approach to automatically adapt Mixed Reality interfaces to the current context, for example, cognitive load, task and environment. We leverage combinatorial optimization to decide when, where and how to display virtual elements. For tasks with low cognitive load, our system displays more elements and in more detail (left). Increased cognitive load leads to a minimal UI with fewer elements at lower levels of detail.

ABSTRACT

We present an optimization-based approach for Mixed Reality (MR) systems to automatically control when and where applications are shown, and how much information they display. Currently, content creators design applications, and users then manually adjust which applications are visible and how much information they show. This choice has to be adjusted every time users switch context, i.e., whenever they switch their task or environment. Since context switches happen many times a day, we believe that MR interfaces require automation to alleviate this problem. We propose a real-time approach to automate this process based on users' current cognitive load and knowledge about their task and environment. Our system adapts which applications are displayed, how much information they show, and where they are placed. We formulate this problem as a mix of rule-based decision making and combinatorial optimization which can be solved efficiently in real-time. We present a set of proof-of-concept applications showing that our approach is applicable in a wide range of scenarios. Finally, we show in a dual-task evaluation that our approach decreased secondary tasks interactions by 36%.

INTRODUCTION

Mixed Reality has the potential to transform the way we interact with digital information. By blending virtual and real worlds, it promises a rich set of applications, ranging from manufacturing and architecture to interaction with smart devices. By their nature, Mixed Reality (MR) interfaces will be context-sensitive: since users are no longer bound to a particular location, MR systems need to adapt to a variety of environmental conditions (e.g., indoor vs outdoors), external (e.g., current task) and internal states (e.g., current concentration level). This inherent context-awareness does, however, pose significant challenges for the design of MR systems: Many UI decisions can no longer be taken at design time but need to be made in-situ, depending on the current context.

Consider the following example. While reading a demanding text, which causes high cognitive load, a user would only want minimal visual augmentations and only those that correspond to the immediate task. However, simply changing location and switching to another activity, such as moving to the living room to take a break, completely changes the user's capacity to process information and the need for additional information in the field of view. With existing approaches, users need to manually adjust the information they see: their unread emails, latest news, or a YouTube video. Switching tasks again (e.g., to a task with medium cognitive load) would require to re-adjust the type and level of detail (LOD) of visible applications. Users have to manually adjust this *every time* they switch contexts, i.e., experience changes in cognitive load, task, and environment, which happens many times a day. This process is cumbersome, especially if applications become embedded in space and in settings with reduced input bandwidth (i.e., without access to mouse and keyboard). We argue that an automated process is necessary to assist users in switching contexts and adapt which applications are shown, how much information they show, and where they are displayed.

Current research in automated MR layout has focused on the automatic positioning of virtual elements based on their visibility with respect to users (e. g., OptiSpace [12]), or based on surface geometry (e. g., FLARE [15]). Approaches for AR view management focused on placing contents such that legibility is increased and clutter decreased (e. g., Bell et al. [6]). None of these works, however, take users' context into account, i. e., jointly adapts an interface based on cognitive load, task, and environment. We argue that these aspects play an equally important role, especially for MR interfaces.

In this paper, we present a first step towards dynamic context-aware MR interfaces that automatically adapt to the users' mental workload, their task and current view of the environment. We propose a combination of rule-based decision making and combinatorial optimization as an approach for MR systems to automatically control *when* applications are shown, *where* they should be displayed and *how much information* they should display.

Concretely, the input for our system are 1) individual applications with different LODs designed by content creators, and 2) a simple specification of which applications are relevant when users perform a task (e. g., when working on a document, relevant applications include email, news, messenger and a todo app). At run time, the system takes the users' current cognitive load, estimated via the Index of Pupillary Activity [10], their task, and their environment into account. We quantify those factors and *optimize* the placement, LOD and time of presentation of virtual contents. Our approach also can take device-specific factors such as the field of view into account.

In summary, we contribute an optimization-based approach to automatically adapt MR interfaces based on the user's current context. We mathematically formulate the relevant factors and constraints in a multi-step optimization process that can be solved efficiently in real-time using rule-based decision making and integer linear programming. We show in an evaluation that our approach is feasible and that our system decreases the number of necessary interactions by 36% in a dual-task scenario. The evaluation was performed in VR to avoid confounding effects of current MR display hardware. We believe that our approach generalizes to MR interfaces, which we show with a set of scenarios, implemented using our real-time optimization-based approach and a camera-based see-through HMD.

APPLICATION SCENARIOS

Before detailing our proposed multi-stage optimization approach, we briefly illustrate a number of application scenarios to clarify the need for and the utility of computational approaches to UI adaptation in the context of MR interfaces.



Figure 2. The email application and its 5 LODs.

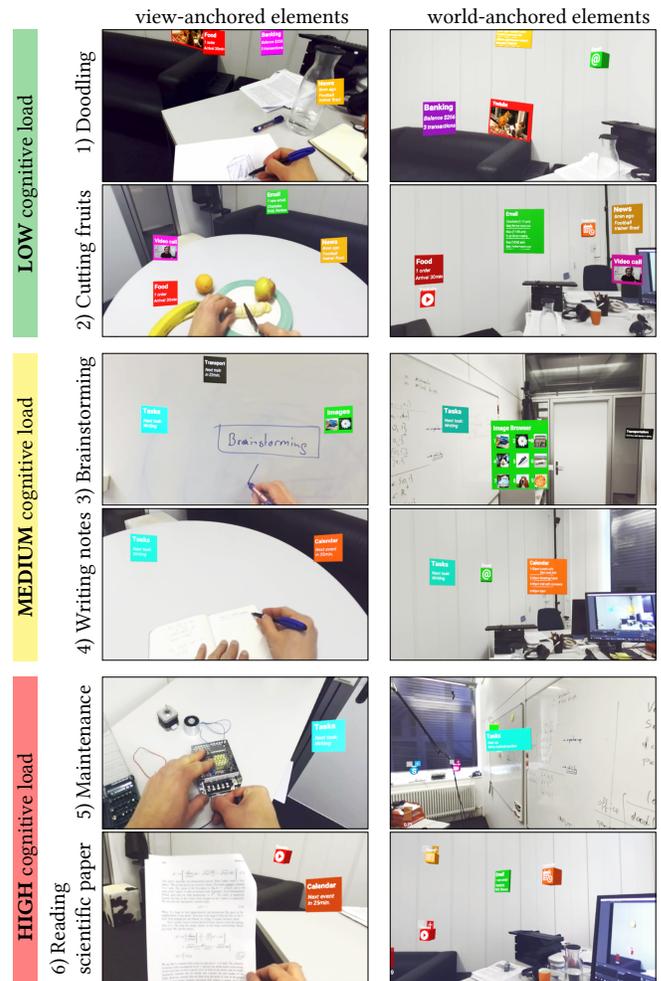


Figure 3. We used a prototypical implementation of our approach to implement 6 scenarios where users experience different levels of cognitive load, different tasks and carry these out in different locations. Thus they are presented with different applications and with varying detail.

A prototypical implementation of our approach is used to adapt a MR interface in various scenarios, see Figure 3. We designed 12 applications (e. g., email, task list, image browser). Each application has 4 to 5 subviews (*LODs*, see Figure 2). In its default state, only an icon (a cube in 3D space) is visible. To automatically adapt the UI to the task we map the frequency of usage of the applications on a scale from 1 (used rarely) to 7 (used always) to a set of tasks (e. g., task "office work" for scenarios 3,4 and 6; frequency of usage for email: 6, tasks: 5, calendar: 5, video call: 4, YouTube: 2).

During tasks such as doodling and cutting fruits (scenarios 1 and 2), users experience a low cognitive load. Our system estimates this and therefore decides to display more information (e. g., YouTube or banking). All applications have, in their initial state, a user-defined position in the environment (i. e., *world-anchored*). If the system determines that an application should be shown but is currently not visible due to viewing angle or occlusion, it is displayed *view-anchored*, i. e., as an heads-up display item. In scenario 2, for example, the green email application is view-anchored, and contains the information that there is one new email (Figure 3, left). Turn-

ing towards the location in space triggers a transition from view-anchored to world-anchored (Figure 3, *right*). Since the available space increased, it is displayed with higher LOD, in this case showing the subjects and senders of the last 3 emails.

During brainstorming and writing notes (scenario 3 and 4), users experience a medium cognitive load. Therefore, fewer elements are displayed. For brainstorming, only 3 applications with low LOD are displayed view-anchored (Figure 3, *left*). The user then turns towards the image browser, which allows our optimization to show more information, i. e., more images (Figure 3, *right*). When performing maintenance on electronics or reading a scientific paper (scenario 5 and 6), users experience high cognitive load. Our system therefore only displays the most frequently used applications (tasks application for maintenance; calendar for reading) and with low LOD to avoid increasing users already high cognitive load.

Manually adjusting the 12 applications in these different tasks would be cumbersome for users. Similarly it is difficult for designers to predict user's cognitive workload during the different tasks and specify the corresponding UI. Our approach automates this process so that content creators can focus on the design of UI elements and end-users do not need to adjust the UI every time they switch their context.

BACKGROUND AND RELATED WORK

Adaptive Mixed Reality

In the context of Mixed Reality there exists several challenges related to the display of virtual content that have been addressed with computational methods. Several works address the problem of automatically determining *where* to place virtual objects in the real world. Fender et al. measured user behavior and object positions to infer the positions of objects for spatial augmented reality [12] and the position of displays [13]. Nuernberger et al. inferred constraints from the physical world to place virtual elements automatically [36]. Gal et al. solved a constraint-satisfaction problem to generate layouts of virtual content based on properties of the physical environment [15]. These approaches can be seen as instances of the view management or label placement problem, addressed by, for example, Bell et al. [6], Azuma et al. [2], Rosten et al. [42], Grasset et al. [18], and Madsen et al. [30].

Few works investigate the problems of *how* virtual content should be shown to the user. Julier et al. performed information filtering based on users' location and task using a region-based approach [27]. DiVerdi et al. proposed to use different LODs of content presentations as a basis for adaptive AR, a concept we also use for our approach [9]. Tatzgern et al. used hierarchical clustering to control the number of labels that are presented to users [46]. Ghouaiel et al. proposed adapting virtual contents based on the environment (illumination, target distance and noise) [16].

In our work, we are jointly concerned with *when*, *where* and *how* to display content. We make these decisions based on users' current cognitive load, and knowledge about their task and environment. Aforementioned work could be used as a pre-processing step to our approach to automatically determine the possible locations to place virtual elements.

Researchers have made different observations on the impact of MR systems on mental workload. Grubert et al. [19] found no impact using mobile AR in an industrial setting, while Tang et al. [45] found that the use of AR increased the mental effort of assembly line workers. In contrast, the use of virtual objects was found to play an important role to reduce subjective workload in collaborative settings where they could be used as spatial cues [34]. Similar to Grubert et al. [20], we argue that context-awareness is a key feature for an elevated user experience. Our work presents a first step towards a holistic system that also incorporates user's mental state.

UI Optimization and adaptation

In the context of mobile UIs and desktop computing, there is a vast range of work calling for and proposing context-aware user interfaces that adapt to the user and their environment (see e. g., [8, 29, 33] for overviews). However, many approaches are based on handcrafted rules and heuristics. Combinatorial optimization has been proposed as a general purpose method to automatically generate and adapt UIs (cf. Oulasvirta [37, 39]) and input devices such as keyboards [11, 28]. For example, Gajos et al. [14] and Sarcar et al. [43] have used optimization methods to adapt desktop and mobile interfaces, respectively, to a user's abilities to interact with a UI (e. g., [14, 43]). However, current approaches are mostly concerned with design-time optimization, customizing a UI once before the user interacts with it. In constantly changing contexts such as MR, a more dynamic approach is needed that continuously adapts UIs in real-time. Park et al. [40] proposed Adam, a system to automatically adapt distributed user interfaces at run-time based on features such as device capabilities and user permissions. Similar to their work, we use an integer linear programming approach to adapt an MR interface but we focus on taking into account users' cognitive load, their task, and their current environment.

Cognitive load estimation

Inferring users' current mental state, e. g., workload or cognitive load, and using this knowledge to adapt interfaces and devices has been a long-standing challenge in HCI. Cognitive load theory (cf. Sweller [44]) states that humans only have a finite amount of cognitive processing capacity. We refer readers to Hollender et al. [25] for a review of cognitive load theory and its applications in HCI. Existing approaches exploit this knowledge e. g., to schedule interrupts (e. g., Bailey et al. [3]) or to adapt user interfaces (e. g., Yuksel et al. [47]). There are three general ways to measure cognitive load: subjective measures such as the NASA TLX [22], performance metrics such as response time, or physiological measures, such as electromyography, skin conductance and respiration (cf. [23, 21]). We use pupil dilation, which Hess and Polt [24] found to be connected to mental difficulty of tasks, and can be measured without instrumenting users (other than tracking their eyes) and computed in real time. Since pupil dilation is confounded with ambient light (cf. [41]), several metrics exist that do not measure absolute size but change in size, such as the Proprietary Index of Cognitive Activity et al. [31, 5], and the Index of Pupillary Activity (IPA) by Duchowski et al. [10], which we use in our work. Our pipeline, however, would be able to incorporate other approaches if desired.

OPTIMIZATION APPROACH

Given a set of UI elements in an MR environment, our goal is to automatically determine each element’s *visibility*, *the amount of information it provides* (LOD) and its *placement* (in the environment or a position within the user’s field of view), at any point in time. Our objective is to maximize the usefulness of the UI while taking into account the user’s view of the environment and an estimation of their current cognitive load. We propose a 3-step process that uses a mix of rule-based decision making and combinatorial optimization. For all elements $e \in E = \{1, \dots, n\}$, our approach first determines if elements should be anchored relative to the user’s view or to the world. It then decides their visibility and LOD for all elements by solving a integer linear program. In a third step, we then determine the placement of elements anchored in the user’s view using a greedy approach to maximize their legibility. In the following, we first describe the inputs and parameters needed by our process and then formulate the optimization problems and constraints.

Inputs

Our approach requires two main sources of input: application-specific parameters which are given by content creators once before the system is used, and context-specific parameters that are determined during runtime (see Table 1).

Application-specific input supplied by content creators

Each element e requires a set of specifications before its visibility, LOD and placement can be determined. $D_e = (1, \dots, m_e)$ denotes the list of subviews that provide different *levels of detail*, where m_e denotes the number of specified subviews for element e . Each subview $d_e \in D_e$ has an associated cognitive cost $c_{e,d_e} \in [0, 1]$, which is an approximation of how much cognitive strain a subview puts on the user. For an individual task t , p_e^t denotes the usage frequency of e . Note that while specifying the frequency of use per LOD would be a straightforward extension of our approach, this would require content creators to set more parameters, which we chose to avoid.

u_{e,d_e}^t denotes the utility of e at the LOD d_e . It approximates how much (useful) information a subview holds. For simplicity, we chose the maximum utility to be similar to the elements frequency of use p_e^t , normalized by the sum of p^t of all elements. For each LOD, the utility is then the ratio to the element’s maximum utility. As an example, a subview with LOD 0 might be an application icon, having a rather low utility. At the highest LOD, an application could contain a list of the 10 recent emails and subjects, yielding a high utility, but also a higher associated cognitive cost.

We provide a simple interface for content creators to specify these parameters on a scale from 1 to 7, which is normalized and used as input. This is done once per application and task. Users do not have to provide any input, but can adjust the position and LOD of applications with dedicated buttons during run time. Our algorithm then takes these as constraints into account. This manual procedure could be replaced with an automatic analysis. Note that the focus of our approach is not to automatically infer user’s current task, nor determining a mapping between the task and a virtual element e . There exist

APPLICATION-SPECIFIC INPUT (SPECIFIED BY CONTENT CREATOR)

Parameter	Description
$E = (e_1, \dots, e_n)$	All virtual elements
$n \in \mathbb{Z}^+$	Number of virtual elements
$D_e = (d_1, \dots, d_{m_e})$	Available LODs for element e
$m_e \in \mathbb{Z}^+$	Number of LODs for element e
$c_{e,d_e} \in [0, 1]$	Cognitive cost of the LOD $d_e \in D_e$ of element e
$u_{e,d_e}^t \in [0, 1]$	Utility of the LOD $d_e \in D_e$ of element e during task t
$p_e^t \in [0, 1]$	Usage frequency of element e during task t

CONTEXT-SPECIFIC INPUT (DETERMINED BY SYSTEM)

Parameter	Description
$v_e \in \{0, 1\}$	Determines if an element e can be seen in the user’s current view of the environment
$L_{est} \in [0, 1]$	The estimated cognitive load of a user at the time of optimization
$L_{max} \in [0, 1]$	The maximal cognitive load capacity of a user

Table 1. Description and ranges of input parameters.

approaches for human activity recognition (see [1] for a review) which would plug into our system. A mapping between current task and application importance could be gathered using data-driven approaches (cf. [4]), and the perceptual costs of individual LODs could be determined using the Aalto Interface Metrics [38]. Our main focus is on developing a flexible and general approach to MR online UI adaptation. Connecting this with more advanced sources of information is a natural extension of our work.

Context-specific input

At run time the process continuously determines if a UI element is visible in the user’s current view of the environment and estimates users’ current cognitive load. We determine if an element e is visible to the user in their current view of the environment, denoted by v_e , as follows:

$$v_e = \begin{cases} 0, & \text{if } e \text{ is occluded,} \\ 0, & \text{if } \omega_e > f, \\ 1 & \text{otherwise} \end{cases} \quad (1)$$

ω_e is calculated as the angle between the user’s current position, and the element’s position in space. $f \in \mathbb{R}$ denotes the diagonal field of view of the device. Occlusion is checked through ray casting from the user’s point of view to the individual elements and intersecting the ray with the virtual environment (in VR) or the acquired geometry of the environment (in AR using devices with front facing depth camera).

To estimate the *cognitive load* L_{est} of the user, we compute the Index of Pupillary Activity (IPA) [10]. The IPA measures the frequency of changes in pupil diameter, which is an indicator for cognitive load. We adapted the implementation provided by Duchowski et al. as detailed in the Implementation section.

In addition to continuously estimating the current cognitive load, we take into account an overall cognitive capacity L_{max} , which is a constant per user. In a pre-study with 5 users, we observed IPA values between 0.05 for tasks with low cognitive load (watching classical music video) and 0.50 for task with high cognitive load (counting backwards in steps of 17, cf. [10]). Based on these tests we set $L_{max} := 0.6$. This factor could be refined by a user-specific calibration in the future. In our experiments and the evaluation, however, using these values yielded good results.

Step 1: View- or world-anchored

For each element e we first decide if it should be view- or world-anchored, captured by the binary decision variable $z_e \in \{0, 1\}$. We first determine the elements' visibility v_e by the user in their current field of view, as described above. If $v_e = 1$, the element is decided to be world-anchored ($z_e = 0$), otherwise it is view-anchored ($z_e = 1$) and will be placed relative to the user's field of view.

Step 2: Visibility and level of detail

In the second step, our goal is to decide if a UI element is displayed and at which LOD. We use integer linear programming to maximize the utility of the UI for the user. The binary decision variable $x_e \in \{0, 1\}$ captures if an element e is displayed or not. $y_{e,d_e} \in \{0, 1\}$ denotes if element e has LODs d_e . Our goal is then to display those elements that are frequently used (p_e) and to maximize their utility u_{e,d_e} . This can be formulate as follows:

$$\max \sum_{e=1}^n \sum_{d_e=1}^{m_e} x_e y_{e,d_e} (p_e + u_{e,d_e}) \quad (2)$$

The usage frequency and utility are calculated for the current task t of the user but for simplicity we drop the superscript. The following constraints restrict the space of feasible solutions and take into account the current cognitive load of the user in order to decide if an element should be displayed and at which LOD.

Cognitive load constraint

Our goal is not only to maximize the usefulness of the displayed elements, but to constrain the cognitive load of users when interacting with an MR interface. Therefore we introduce the constraint that the cognitive cost L_{vir} induced by all displayed virtual elements together with the estimated cognitive load of the user L_{est} under the current task and environment cannot exceed the overall cognitive capacity L_{max} and should leave a minimum remaining capacity α .

$$L_{est} + L_{vir} \leq L_{max} - \alpha \quad (3)$$

where

$$L_{vir} = \sum_{e=1}^n \sum_{d_e=1}^{m_e} x_e y_{e,d_e} c_{e,d_e} \quad (4)$$

L_{vir} denotes the sum of the costs of the displayed elements depending on their LOD. In our implementation we set $\alpha := 0.1$, based on the results of aforementioned pre-study.

Level of detail constraints

To ensure continued usefulness of individual apps, we add several constraints related to the LOD an element is displayed with. The first constraint, formulated as,

$$\sum_{d_e=1}^{m_e} y_{e,d_e} = 1 \quad \forall e \in \{1, \dots, n\} \quad (5)$$

ensures that each element is displayed with only one LOD.

If a user is currently interacting with an application, it would be disturbing to automatically change the LOD. We therefore introduce a regularizer that fixes the LOD if the time Δ_{e,d_e} since a user has been last interacting with an element is below a threshold γ (in our case 60 seconds). This can be formulate as

$$\Delta_{e,d_e} < \gamma \Rightarrow y_{e,d_e} = 1 \quad \forall e \in \{1, \dots, n\}, d_e \in \{1, \dots, m_e\} \quad (6)$$

If a user has been interacting with multiple LODs (e. g., by manually decreasing or increasing), we pick the last one they have interacted with. Note that in the case where a user closes an element we can use a similar constraint but require $x_e = 0$.

Finally, we constrain the LOD of elements that are view-anchored. Continuously displaying an element in the user's field of view typically means that less information should be displayed to not overload users (e. g., by displaying a full news article close to user's focal view). This is captured by the following inequality constraint

$$z_e = 1 \Rightarrow y_{e,d_e} = 0 \quad \forall e \in \{1, \dots, n\}, d_e \leq 2 \quad (7)$$

which states that in the case where an element is view-anchored, it cannot take levels of detail which are greater than 2, assuming they are ordered by amount of information.

Note that in our prototype implementation, even if the optimization decides that an element is not visible, we still display a simple icon representing the application, so that users can access it if necessary. If applications could be launched using a menu or controller button, for example, this would not be necessary and elements could be hidden completely.

Step 3: Placement within view

The final step determines the placement of elements that are anchored in view. We assume a predefined set of slots in the field of view which can host elements at different levels of detail as determined in the previous step. Our goal is to show the elements with the highest usage frequency and utility at the best slots. Therefore, we assign a score to each of the slots, based on their quality in terms of their position and legibility. Formally, the quality q_s of a slot s is defined as

$$q_s = e^{-10 \cdot (\sqrt{s_x^2 + s_y^2} - 0.5)^2} \quad (8)$$

$s_x, s_y \in [-1, 1]$ denote the (x,y) position of the center of the slot relative to the headset, e. g., the point (0,0) is the center of the user's view, (1,1) is the top right corner of the headset. We chose this function to balance keeping elements out of the

user’s central field of view to avoid occlusion, with not placing elements too far in the periphery, where contents tend to look blurry due to current hardware limitations. A visualization of the distribution of scores can be seen in Figure 4. Assignment of virtual elements to slots is performed in a greedy manner, with elements with higher utility assigned to slots with higher quality. If an element has previously been assigned to a slot, it will be re-assigned to it to exploit user’s spatial memory.

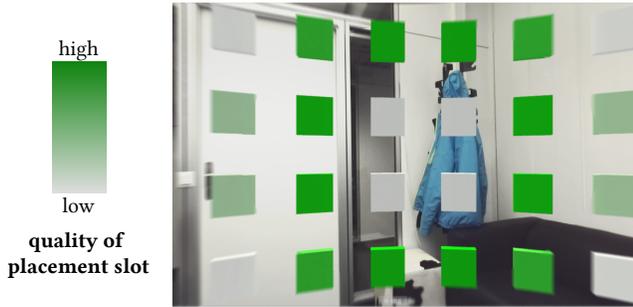


Figure 4. Quality of the view-anchored placement slots. The quality function can be adapted based on hardware capabilities or user’s preferences (e. g., prefer left over right).

IMPLEMENTATION

For delivering a see-through MR experience, we used a HTC Vive Pro VR headset, augmented with a front-facing RGB-D camera (Stereolabs Zed Mini). We decided against using an optical see-through AR headset (e. g., Microsoft HoloLens) because of their limited field of view (usually around 30° to 45°). The combination of VR headset and RGB-D camera resulted in a field of view of approximately 90° (H) x 60° (V). For gaze tracking, the headset was equipped with an integrated eye tracker (Pupil Labs HTC Vive Binocular Add-on, 120 Hz per camera).

We implemented our software in Unity 2018, which allows for using it in AR and VR scenarios. Gurobi 8.1 is used to solve the integer program formulated above. We use Python 3.6 to interface with the Pupil SDK and to compute the Index of Pupillary Activity. The calculated IPA is sent to Unity through a local socket. Inputs for the optimization are provided in Unity through a custom plugin. Content creators can specify tasks and set the frequency of use for individual virtual elements (see Figure 5).

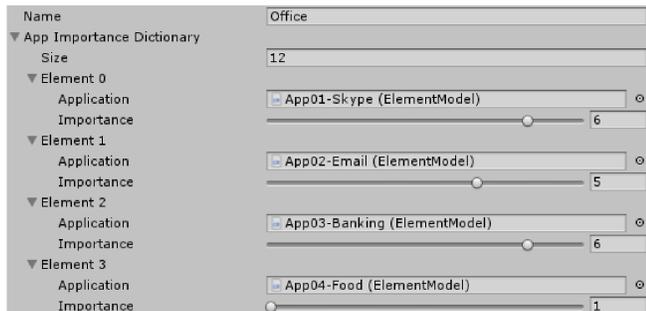


Figure 5. Content creators specify the frequency of use of individual elements, in this example for the task *office work*.

Temporal smoothing

In our early tests, we found that our optimization tended to jump between solutions, meaning that elements appeared and disappeared quickly, depending on the current context. This behavior is typical for discrete optimization formulations such as the integer linear program presented above. To alleviate this challenge, we introduce a temporal smoothing that ensures improved transitions between solutions. A state manager stores the state of the optimization (i. e., visibility, LOD and placement of the virtual elements) every time the optimization completes. Only if a user’s cognitive load changes by more than 5%, the optimization is restarted. Similarly, if a user changes position or viewing direction, the optimization restarts. To avoid that virtual elements change their position or appearance every time a user moves, we delay the optimization for 60 seconds and only restart it if the change in state is still present. This avoids situations where users turn their head to look up information and then resume with their task, only to find that their interface has changed.

Computing the IPA

We base our implementation on the work of Duchowski et al. [10]. We chose this metric since it can be computed without instrumenting users (apart from an eye tracker in the MR headset), as well due to the possibility to compute it during interaction. Initial tests showed that a 60 seconds sliding-window provided a good balance between real-time computation and resilience to noise. We found that shorter sliding windows (e. g., 30 seconds) yielded a less reliable signal with more fluctuations. Longer sliding windows of up to 3 minutes did not provide visible improvements over 60 seconds, while introducing an even longer delay. To account for noise between measurements, we apply exponential smoothing on the computed IPA with $\alpha = \frac{1}{f}$, where f denotes the sampling rate of the eye tracker, in our case $f = 120$ fps, i. e., $\alpha = 0.0083$.

Computational considerations

In our initial evaluation, detailed in the next section, we found that the optimization of 12 elements with four to five levels of detail each, took 7.1 ms on average ($SD = 4.9$ ms) on a commodity gaming PC (Intel Core i7-8086K, 4GHz with 6 cores, 32GB Ram, Windows 10, NVIDIA GeForce GTX 1080 Ti), allowing for real-time interaction. During the experiment, the optimization ran approximately 1200 times in total. We note that the integer program formulated above is an instance of the knapsack problem, which is known to be NP-hard [32]. Thus, to ensure that our approach also runs in real-time for more elements, we set out to test the scalability of our approach. We populated the optimization with 30, 60, 100 and 200 elements, respectively, with 4 levels of detail per element. Each element was initialized with random values for frequency of use, utility, and cost. We then ran the optimization 1000 times for each set, each time with a different randomly chosen cognitive load. For 30 elements, the optimization took $M = 6.14$ ms ($SD = 1.29$ ms), and for 200 elements 43.55 ms ($SD = 4.78$ ms), with approximately linear growth for 60 and 100 elements. This indicates that even with 200 elements and their total of 800 levels of detail, our optimization can easily run at interactive rates.

EVALUATION

To explore the benefits and limitations of the automatic adaptation, we analyzed its performance and perceived usability in an empirical user study. In our experiment, 12 participants subsequently performed three tasks exhibiting different cognitive load. During each task they were asked to perform a secondary task (verbally answering questions) in which they relied on information provided by the elements placed in their environment. Participants performed the tasks with and without our system enabled. In terms of research questions, we were interested in 1) how our approach influences task performance, 2) how our system influences perceived usability and distraction and 3) if participants will be able to predict the actions of our system.

Participants

We recruited 12 participants between 23 and 36 years ($M = 29$ years, 4 female, 8 male) from a local university. All participants had little to no experience using AR systems. 3 participants rated themselves as experienced users with VR systems, all others reported little to no experience. Participants had corrected or corrected-to-normal vision based on self reports. No participants reported elevated susceptibility for motion sickness when queried using the Motion Sickness Susceptibility Questionnaire Short-form (MSSQ-Short) [17].

Study design

We used a within-subject design with two independent variables: *Task* with 3 levels (*Count17*, *Count2* and *IconSearch*, inducing different cognitive loads) and *Method* with 2 levels (*Optimized* and *Manual*), yielding 6 conditions. In each condition, participants were asked to complete a secondary task, which involved answering questions posed via computer-generated audio. As dependent variables, we measured primary and secondary task performance (time and errors), and number of interactions with the UI elements. For each level of *Method*, participants performed each primary task once for 4 minutes. The order of *Task* and *Method* was counterbalanced across participants, but *Task* order was kept the same within the *Method* conditions of individual participants.

Primary tasks

For *Count2* and *Count17*, participants were asked to count backwards in steps of 2 and 17, respectively, starting from a random 3-digit number between 600 and 800 (cf. Duchowski et al. [10]). Participants saw the starting number as a virtual element at a fixed position and were asked to input their responses on a number pad with the VR controller (Figure 6, left). For *IconSearch*, participants were presented with a set of 30 randomly selected icons and indicate if a target icon was contained in the set (Figure 6, right) by pressing a virtual button. The target icon and the icon set were changed after every trial to avoid learning effects. The primary tasks were designed to yield different cognitive loads, from high (*Count17*) to low (*Count2*). Switching tasks also meant switching physical position, since each task was located at a different position, as shown in Figure 6. Participants were instructed to *perform the task as fast as possible without making any errors*.

Secondary task

As a secondary task, participants were asked questions (predefined but unknown to them). In each *Task*, 9 questions were randomly selected from a set of 20 questions. They were related to 4 to 5 different applications specified as relevant for the current primary task, and the answer could be found by opening the correct LOD. Questions were posed verbally by the software at a randomly-set interval between 25 to 30 seconds. As an example, participants were asked "When did Sam write you the last email?", which could be answered when opening the second LOD of the email application. Participants were instructed to *answer all questions in a timely manner*.

We chose this task since it closely resembles the spontaneous recall of information or an interrupt by a third party. In such situations, users have to consciously make an effort to look for information in the interface. Depending on the currently visible UI elements and their LOD, users have to perform more or less interactions to gather the information, and may need longer or shorter time for completing this task.

We did not expect to see large differences in quantitative task performance, since we believe that the cognitive load and performance is largely dominated by the primary task, and less by occasional actions for the secondary task.

Method

We compared our optimization-based approach (*Optimized*) against a manual approach (*Manual*). To perform the secondary task, in both cases, participants could increase or decrease the levels of detail of applications using two dedicated buttons on top of every virtual element. In the *Manual* condition, all applications started in their lowest LOD, i. e., only the icon was shown. Applications that were opened by participants were shown view-anchored in the selected LOD when they would not be visible in the user's current view of the environment. In the *Optimized* condition, the visibility of elements anchored to the users' view and their LOD was automatically chosen by our optimization process. As input to the process, we defined the usage frequency of the virtual elements for each task p'_e roughly consistent with the number of questions related to that element. No other study-specific input was provided to the process. All parameters were kept constant throughout the experiment for all participants.



Figure 6. Environment of the experiment. In the center are the 12 applications needed to complete the secondary task. Left and right are two of the primary tasks, counting and icon search, respectively. Note that only one primary task was visible to participant at a time.

Apparatus

We used the same apparatus as described in the Implementation section, except the front-facing depth camera. The camera-based pass-through MR solution introduces a camera-to-display lag of approximately 60 ms, which we found to be uncomfortable for participants. This delay is purely due to the camera setup, not because of our optimization, and is not present when using the VR headset. We chose a pure VR environment since it offers a lag-free viewing experience with large field of view that is fully controllable and consistent across participants. The experimental software was written in Unity 2018. Secondary task questions were posed using the built-in Windows 10 text-to-speech module. The experiment was performed in a distraction-free experimental room (size 3 × 3 meters), with a play area in VR of 2.5 × 2 meters. This was sufficient for participants to move freely without having to worry about colliding with the environment.

Procedure

After signing a consent form, participants completed the demographic questionnaire and the MSSQ-Short. They could familiarize themselves with the tasks and environment in a 5 minute training. When they stated to feel comfortable performing the tasks, the experiment started. After completing the 3 Tasks in one Method condition, participants completed the Nasa-TLX [22] and the SUS questionnaire [7], extended with questions specific to the experiment. All questions were answered on a 5-point scale from low (1) to high (5). After completing all conditions, participants filled in a post-experiment questionnaire and were debriefed. The full study lasted ca. 60 minutes per participant.

Data collection and analysis

Data was collected by logging all events in Unity. We defined primary task completion time as the time from appearance of the trial until it was completed, either by inputting the correct number (*Count17* and *Count2*) or by deciding if an icon is part of the displayed set (*IconSearch*). We removed trials where participants made an error, between 5 and 24 per participant, overall 6% of the data. For secondary task completion time, answer timestamps were recorded when the experimenter pressed a button. We counted the number of interactions performed to increase the LOD to reveal the necessary information for answering the question. To not bias the analysis in favor of *Optimized*, we did not include interactions performed to close an application (in *Manual* an opened application was shown view-anchored until closed).

To analyze the dependent variables, we performed a series of Kruskal-Wallis tests to identify main effects, and Wilcoxon signed-rank tests with Bonferroni correction ($\alpha = 0.017$) for post-hoc comparisons. Differences between questionnaire answers were tested for significance using Wilcoxon signed-rank tests (Bonferroni corrected $\alpha = 0.017$).

Results

We found that participants required $M = 36\%$ less interactions to complete their secondary tasks in the *Optimized* than in the *Manual* condition. The primary task dominated participants' performance, which is exhibited in the fact that Method

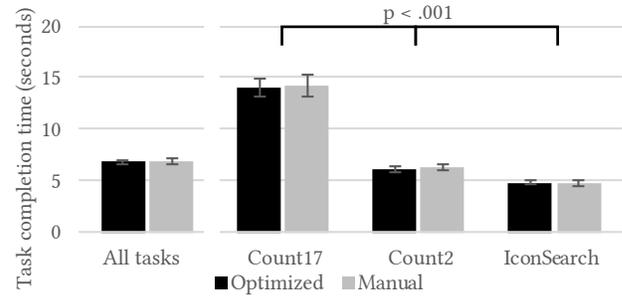


Figure 7. Primary task completion time for all conditions. Error bars indicate standard error.

did not significantly influence the other dependent variables (primary and secondary task performance). *Optimized* and *Manual* exhibited similar task performance, showing that our system performed similar to a traditionally designed UI. In the *Optimized* condition, participants rated themselves to be less distracted, found little inconsistency in the system, and reported that they could predict the actions of the system well.

Performance

Figure 7 shows the primary task completion times and statistical significant differences between Tasks ($\alpha = .05$). We found a main effect of Task ($\chi^2 = 555.994$, $p < .001$), but no main effect for Method ($p = 0.232$). Post-hoc tests revealed that primary task completion time was highest for *Count17*, followed by *Count2* and *IconSearch* (all $p < .001$). For secondary task time, we did not find a main effect for neither Task nor Method. In all conditions, participants took on average between 9.27 sec ($SD = 4.32$ sec) and 10.54 sec ($SD = 3.63$ sec). Given that the timestamps were recorded manually, this falls within the margin of human error when pressing a button.

We found a significant difference in the number of interactions between Tasks ($\chi^2 = 10.37$, $p = .006$) and Methods ($\chi^2 = 24.24$, $p < .001$). As shown in Figure 8, *Optimized* yielded a statistically significant lower number of interactions in all conditions (all $p < .01$). For the *Manual* condition, participants increased the LOD 57.2% of the time, and decreased the LOD 42.8% of the time. This suggests that participants preferred showing less information to avoid clutter. Otherwise they would have kept the application at a high LOD in anticipation of upcoming questions.

To evaluate the accuracy in showing the correct LOD, we compared the target LOD (i. e., minimum LOD to find the answer to the secondary task) to the LOD presented by our approach. The result is shown in Figure 9. Our system generally performs better for lower LOD targets. For example, when the answer can be found in LOD 2, the optimization displays the correct LOD 50% of the time and shows too little information 37% of the time, requiring participants to manually increase the level of detail by 1. For higher LOD, our system tends to act conservatively and shows lower-than-needed LODs. In addition to the target application, our optimization shows $M = 2.21$ LODs ($SD = 1.47$) of other applications across all tasks and target LODs. This is because it only knows the general task of participants (i. e., a set of potentially relevant applications) but not the specific question posed during the study.

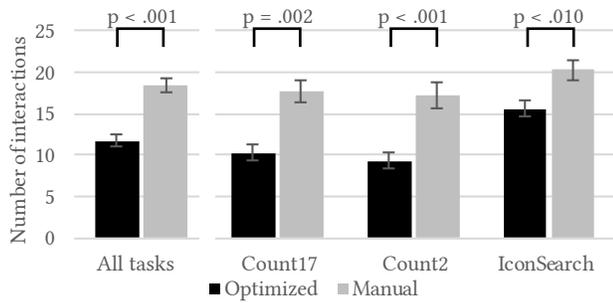


Figure 8. Number of interactions required to complete the secondary task. Error bars indicate standard error.

Questionnaire

Participants rated *Optimized* ($M = 1.91$, $SD = 1.00$) less distracting than *Manual* ($M = 3.00$, $SD = 0.95$), $Z = -0.846$, $p = .052$. On the question "I thought there was too much inconsistency in this system.", participants rated *Optimized* ($M = 1.16$, $SD = 0.37$) lower than *Manual* ($M = 2.42$, $SD = 1.32$), $Z = -1.00$, $p = .021$. However, the differences did not reach statistical significance ($\alpha = .017$). On the question "I could predict the actions the system was performing.", participants rated *Optimized* ($M = 4.08$, $SD = 0.95$) higher than *Manual* ($M = 2.42$, $SD = 1.19$), $Z = 0.359$, $p = .010$. No significant differences were found between *Methods* for any other question. In general, participants rated themselves reasonably successful for performing the tasks for *Optimized* ($M = 3.58$, $SD = 0.51$) and *Manual* ($M = 3.66$, $SD = 0.65$) on a scale from failure (1) to perfect (5), $p = .77$.

Discussion of results

While participants completed the secondary task with significantly fewer interactions using our approach, no difference was present for primary and secondary task completion time. This shows that the tasks dominated the cognitive load and performance, rather than the interaction with elements. This is not unexpected. We believe that while our approach generally has the potential to improve this type of interactions, it might be more suitable for a longer lasting adaptation of an interface after a context switch. Participants were able to anticipate new questions, and were therefore more interested in a very fast interaction technique than a system that decreased their overall cognitive load. They acknowledged, however, that our system has the potential to decrease distraction. Participants also commented that the view-anchored items were generally helpful, however were often blurry, thus hard to read, due to hardware limitations. Since for most participants the baseline and our system were novel, and we never introduced the system as "made to decrease distraction", we believe that this should not be attributed to a good participant effect. Our optimization approach does, however, require more longitudinal investigation, potentially having participants using the system throughout the day while performing multiple long running tasks. Current hardware, however, with limited field of view and prohibitive weight and tracking restrictions does not allow for such true in-the-wild studies. We hope to perform such a longitudinal study on a small number of participants in the future as technology advances.

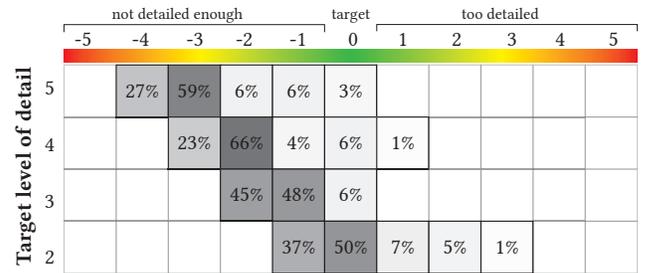


Figure 9. Difference between target LoD for the secondary task and suggestions by our optimization. Our optimization acts rather conservatively and tends to display lower LOD, rather than overshooting and showing too much information.

DISCUSSION, LIMITATIONS AND FUTURE WORK

In this paper, we propose an optimization approach that automatically adapts virtual elements in an MR interface to user's current cognitive load and view of the environment, using knowledge about the task they perform. Our goal was to facilitate the design of MR applications by automating the process of adapting UI elements to the context of users which varies frequently in MR setting. Our optimization process automatically decides which applications to display, where they should be displayed, and how much information they should contain. This allows content creators to focus on the design of UI elements rather than specifying their behavior for any change in the user's context. In the following, we discuss the different aspects of our system in the light of our first exploratory user study and the set of implemented applications.

Automatically placing elements

In an early version of our system, we automatically and continuously computed the best placement of virtual elements anywhere in the world based on their current utility and visibility (similar to HeatSpace [13]). This led to objects constantly changing their position, which would prohibit users exploiting spatial memory and lead to confusing behavior. We believe that automatic placement systems can be useful as a pre-processing step to provide an initial estimate where objects should be placed. Once the elements are placed either manually by users or an automated system, we recommend keeping their position fixed but only changing their visibility if necessary.

Our approach of automatically deciding if an object should be world- or view-anchored provides a balance between quick access to information and predictable location of elements. View-anchored elements are an alternative to automated placement that allow users to access information without disturbing spatial memory even when world-anchored elements are not in the view. Users can then choose to turn to world-anchored objects if they need to gather more information. In our user study, we found that while users appreciated the view-anchored elements, they oftentimes had problems reading the available information, which appeared blurry. Limited visual quality and field of view is a challenge with recent MR headsets, which we hope will be solved in the near future. This would also allow us to further investigate the limits of how much information should be displayed close to user's field of view.

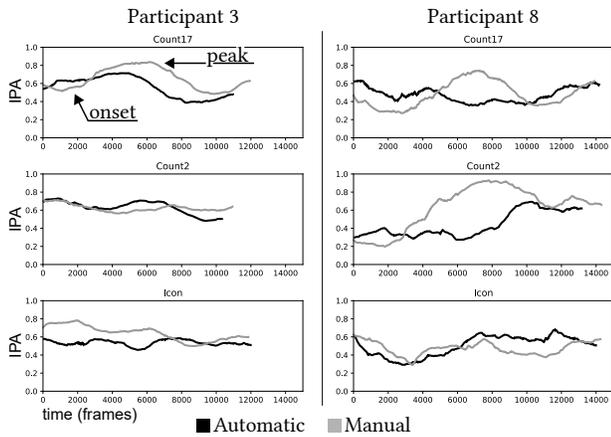


Figure 10. Normalized IPA by condition for two participants of the study with example annotations for onset and peak (top left).

Applications vs. augmentations

The applications we implemented are based on existing applications (e. g., mail, messengers), therefore are more text-focused and rectangular. MR has the potential for more complex augmentations of physical objects, such as recoloring or hiding them. While we believe that our optimization-based approach generalizes to those scenarios, further testing is needed to evaluate its applicability, benefits and challenges. Creating MR interfaces in complex environments with a rich set of applications and augmentations will be even more challenging for content creators when using a manual workflow. We therefore believe that this is also where the strengths of our approach come into play, and that context-aware automatic adaptation is a necessity to make such interfaces feasible.

Cognitive load estimation and user state

In our experiment, the IPA was a good indicator of cognitive load for most participants. However, a behavior that we frequently observed is a peak in cognitive load after about 2 minutes into each condition, and a decline thereafter, as shown for two participants in Figure 10 (similar graphs for all participants can be found in the Appendix, Figure 11). This might be attributed to participants adjusting to the task. Interestingly, we could not find statistically significant differences when comparing the IPA across different *Tasks*. This suggests that either our experimental method saturated the cognitive capacity of participants, or that the IPA can only be used as a relative load indicator with a reliability of several minutes. While for our purposes, the IPA was a useful indicator of cognitive load, in particular within a condition, more investigation is needed to prove the reliability of the IPA during everyday use. During our experiments, the parameters of the cognitive load estimation and the optimization in general were kept constant as described in the Implementation section. While we believe they were sufficiently robust across users for our purposes, further testing is needed to find out if personalized parameters might improve its accuracy.

One challenge of the IPA, and cognitive load estimation in general is their real-time usage. While the IPA can be easily computed in real-time, it relies on data that incorporates e. g., the last 60 seconds. Therefore, sudden events such as

the start of a conversation can only be detected with a delay of approximately 30 seconds, although they will influence cognitive load immediately. This has also been discussed by Hudson et al. [26], who found that the most reliable metric for interruptability are conversations. Therefore, we believe that relying on a sensor fusion approach for cognitive load estimation and interruptability will yield the best real-time results. Our approach easily allows using other real-time estimates of cognitive load (e. g., measured through galvanic skin response such as by Nourbakhsh et al. [35]) if they prove to be more reliable. We hope to investigate this in the future and expand our framework with different sources of cognitive load estimation and user state analysis.

Predictability

For any system that automates how an interface appears in real time, users' ability to predict current and future states is key. We introduced a set of regularizers to avoid sudden changes to the interface when users are interacting with it, as well as avoided automated spatial placement. In our experiment, we found that participants were able to predict the actions of the system at a high rate. We believe that our approach provides a good balance between automation and consistency. The main challenge is that any system that provides more automation essentially has to be an "oracle" and must be able to perfectly predict user's future state and which applications will be needed.

Besides estimating user' current cognitive load, our current system relies on the manual specification of users' current task, which could be replaced by a task or activity estimation approach and automated scene understanding. It is unclear, however, how fine-grained this task estimation has to be, e. g., should the task be more general such as "cooking" or more fine-grained such as "chopping vegetables". We plan to explore this further in the future.

CONCLUSION

We presented a computational approach based on a combination of rule-based decision making and integer linear programming to decide where and when virtual elements should be placed in a Mixed Reality context, and how much information those element should provide. By leveraging information on the user's current environment, task and cognitive load, we automatically adapt MR interfaces, which are highly dynamic by nature, to fit user's current context. Our approach allows content creators to move from a purely manual workflow of content creation, where every situation has to be dealt with separately, to a flexible assignment of task and application. Our system then decides if an application should be displayed, and with which level of detail.

We believe that automatic adaptation in general is the only way to handle a constantly changing context and provide users with useful information and content *at the right time, place and with the right level of detail*. Currently, content creators would have to guess how an interface should behave when a user performs a particular task. Our optimization approach is a first step towards making MR interfaces more usable by moving from a manual workflow of content creation to a process that is computationally supported.

ACKNOWLEDGEMENT

The authors would like to thank Alexandra Ion for her help and feedback, and Seonwook Park for his help with the video. This work was supported in part by the ETH Zurich Postdoctoral Fellowship Programme (ETH/Cofund 18-1 FEL-39) and the ERC Grant OPTINT (StG-2016-717054).

REFERENCES

- [1] J.K. Aggarwal and Michael Sahngwon Ryoo. 2011. Human Activity Analysis: A Review. *ACM Comput. Surv.* 43, 3, Article 16 (April 2011), 43 pages. DOI: <http://dx.doi.org/10.1145/1922649.1922653>
- [2] Ronald Azuma and Chris Furnanski. 2003. Evaluating Label Placement for Augmented Reality View Management. In *Proceedings of the 2nd IEEE/ACM International Symposium on Mixed and Augmented Reality (ISMAR '03)*. IEEE Computer Society, Washington, DC, USA, 66–. <http://dl.acm.org/citation.cfm?id=946248.946790>
- [3] Brian P. Bailey and Shamsi T. Iqbal. 2008. Understanding Changes in Mental Workload During Execution of Goal-directed Tasks and Its Application for Interruption Management. *ACM Trans. Comput.-Hum. Interact.* 14, 4, Article 21 (Jan. 2008), 28 pages. DOI: <http://dx.doi.org/10.1145/1314683.1314689>
- [4] Gilles Bailly, Antti Oulasvirta, Timo Kötzing, and Sabrina Hoppe. 2013. MenuOptimizer: Interactive Optimization of Menu Systems. In *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology (UIST '13)*. ACM, New York, NY, USA, 331–342. DOI: <http://dx.doi.org/10.1145/2501988.2502024>
- [5] Michael Bartels and Sandra P. Marshall. 2012. Measuring Cognitive Workload Across Different Eye Tracking Hardware Platforms. In *Proceedings of the Symposium on Eye Tracking Research and Applications (ETRA '12)*. ACM, New York, NY, USA, 161–164. DOI: <http://dx.doi.org/10.1145/2168556.2168582>
- [6] Blaine Bell, Steven Feiner, and Tobias Höllerer. 2001. View Management for Virtual and Augmented Reality. In *Proceedings of the 14th Annual ACM Symposium on User Interface Software and Technology (UIST '01)*. ACM, New York, NY, USA, 101–110. DOI: <http://dx.doi.org/10.1145/502348.502363>
- [7] John Brooke and others. 1996. SUS-A quick and dirty usability scale. *Usability evaluation in industry* 189, 194 (1996), 4–7.
- [8] Anind K Dey. 2016. Context-Aware Computing. In *Ubiquitous Computing Fundamentals*. Chapman and Hall/CRC, 335–366.
- [9] Stephen DiVerdi, Tobias Höllerer, and Richard Schreyer. 2004. Level of Detail Interfaces. In *Proceedings of the 3rd IEEE/ACM International Symposium on Mixed and Augmented Reality (ISMAR '04)*. IEEE Computer Society, Washington, DC, USA, 300–301. DOI: <http://dx.doi.org/10.1109/ISMAR.2004.38>
- [10] Andrew T. Duchowski, Krzysztof Krejtz, Izabela Krejtz, Cezary Biele, Anna Niedzielska, Peter Kiefer, Martin Raubal, and Ioannis Giannopoulos. 2018. The Index of Pupillary Activity: Measuring Cognitive Load Vis-à-vis Task Difficulty with Pupil Oscillation. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. ACM, New York, NY, USA, Article 282, 13 pages. DOI: <http://dx.doi.org/10.1145/3173574.3173856>
- [11] Anna Maria Feit. 2018. *Assignment Problems for Optimizing Text Input*. Ph.D. Dissertation. Aalto University. <http://urn.fi/URN:ISBN:978-952-60-8016-1>
- [12] Andreas Fender, Philipp Herholz, Marc Alexa, and Jörg Müller. 2018. OptiSpace: Automated Placement of Interactive 3D Projection Mapping Content. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. ACM, New York, NY, USA, Article 269, 11 pages. DOI: <http://dx.doi.org/10.1145/3173574.3173843>
- [13] Andreas Fender, David Lindlbauer, Philipp Herholz, Marc Alexa, and Jörg Müller. 2017. HeatSpace: Automatic Placement of Displays by Empirical Analysis of User Behavior. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology (UIST '17)*. ACM, New York, NY, USA, 611–621. DOI: <http://dx.doi.org/10.1145/3126594.3126621>
- [14] Krzysztof Gajos and Daniel S. Weld. 2004. SUPPLE: Automatically Generating User Interfaces. In *Proceedings of the 9th International Conference on Intelligent User Interfaces (IUI '04)*. ACM, New York, NY, USA, 93–100. DOI: <http://dx.doi.org/10.1145/964442.964461>
- [15] Ran Gal, Lior Shapira, Eyal Ofek, and Pushmeet Kohli. 2014. FLARE: Fast layout for augmented reality applications. In *2014 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. 207–212. DOI: <http://dx.doi.org/10.1109/ISMAR.2014.6948429>
- [16] Nehla Ghouaiel, Jean-Marc Cieutat, and Jean-Pierre Jessel. 2014. Adaptive Augmented Reality: Plasticity of Augmentations. In *Proceedings of the 2014 Virtual Reality International Conference (VRIC '14)*. ACM, New York, NY, USA, Article 10, 4 pages. DOI: <http://dx.doi.org/10.1145/2617841.2620695>
- [17] John F. Golding. 2006. Predicting individual differences in motion sickness susceptibility by questionnaire. *Personality and Individual Differences* 41, 2 (2006), 237–248. DOI: <http://dx.doi.org/10.1016/j.paid.2006.01.012>
- [18] Raphael Grasset, Tobias Langlotz, Denis Kalkofen, Markus Tatzgern, and Dieter Schmalstieg. 2012. Image-driven View Management for Augmented Reality Browsers. In *Proceedings of the 2012 IEEE International Symposium on Mixed and Augmented Reality (ISMAR) (ISMAR '12)*. IEEE Computer Society, Washington, DC, USA, 177–186. DOI: <http://dx.doi.org/10.1109/ISMAR.2012.6402555>

- [19] Jens Grubert, Daniel Hamacher, Rüdiger Mecke, Irina Böckelmann, Lutz Schega, Anke Huckauf, Mario Urbina, Michael Schenk, Fabian Doil, and Johannes Tümler. 2010. Extended investigations of user-related issues in mobile industrial AR. In *2010 IEEE International Symposium on Mixed and Augmented Reality (ISMAR '10)*. IEEE, 229–230. DOI : <http://dx.doi.org/10.1109/ISMAR.2010.5643581>
- [20] Jens Grubert, Tobias Langlotz, Stefanie Zollmann, and Holger Regenbrecht. 2017. Towards Pervasive Augmented Reality: Context-Awareness in Augmented Reality. *IEEE Transactions on Visualization and Computer Graphics* 23, 6 (June 2017), 1706–1724. DOI : <http://dx.doi.org/10.1109/TVCG.2016.2543720>
- [21] Eija Haapalainen, SeungJun Kim, Jodi F. Forlizzi, and Anind K. Dey. 2010. Psycho-physiological Measures for Assessing Cognitive Load. In *Proceedings of the 12th ACM International Conference on Ubiquitous Computing (UbiComp '10)*. ACM, New York, NY, USA, 301–310. DOI : <http://dx.doi.org/10.1145/1864349.1864395>
- [22] Sandra G. Hart and Lowell E. Staveland. 1988. Development of NASA-TLX (Task Load Index): Results of Empirical and Theoretical Research. In *Human Mental Workload*, Peter A. Hancock and Najmedin Meshkati (Eds.). Advances in Psychology, Vol. 52. North-Holland, 139 – 183. DOI : [http://dx.doi.org/10.1016/S0166-4115\(08\)62386-9](http://dx.doi.org/10.1016/S0166-4115(08)62386-9)
- [23] Jennifer A. Healey and Rosalind W. Picard. 2005. Detecting Stress During Real-world Driving Tasks Using Physiological Sensors. *Trans. Intell. Transport. Sys.* 6, 2 (June 2005), 156–166. DOI : <http://dx.doi.org/10.1109/TITS.2005.848368>
- [24] Eckhard H Hess and James M Polt. 1964. Pupil size in relation to mental activity during simple problem-solving. *Science* 143, 3611 (1964), 1190–1192.
- [25] Nina Hollender, Cristian Hofmann, Michael Deneke, and Bernhard Schmitz. 2010. Review: Integrating Cognitive Load Theory and Concepts of Human-computer Interaction. *Comput. Hum. Behav.* 26, 6 (Nov. 2010), 1278–1288. DOI : <http://dx.doi.org/10.1016/j.chb.2010.05.031>
- [26] Scott Hudson, James Fogarty, Christopher Atkeson, Daniel Avrahami, Jodi Forlizzi, Sara Kiesler, Johnny Lee, and Jie Yang. 2003. Predicting Human Interruptibility with Sensors: A Wizard of Oz Feasibility Study. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '03)*. ACM, New York, NY, USA, 257–264. DOI : <http://dx.doi.org/10.1145/642611.642657>
- [27] Simon Julier, Yohan Baillot, Dennis Brown, and Marco Lanzagorta. 2002. Information Filtering for Mobile Augmented Reality. *IEEE Comput. Graph. Appl.* 22, 5 (Sept. 2002), 12–15. DOI : <http://dx.doi.org/10.1109/MCG.2002.1028721>
- [28] Andreas Karrenbauer and Antti Oulasvirta. 2014. Improvements to Keyboard Optimization with Integer Programming. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology (UIST '14)*. ACM, New York, NY, USA, 621–626. DOI : <http://dx.doi.org/10.1145/2642918.2647382>
- [29] Henry Lieberman and Ted Selker. 2000. Out of context: Computer systems that adapt to, and learn from, context. *IBM systems journal* 39, 3.4 (2000), 617–632. DOI : <http://dx.doi.org/10.1147/sj.393.0617>
- [30] Jacob Boesen Madsen, Markus Tatzgern, Claus B. Madsen, Dieter Schmalstieg, and Denis Kalkofen. 2016. Temporal Coherence Strategies for Augmented Reality Labeling. *IEEE Transactions on Visualization and Computer Graphics* 22, 4 (April 2016), 1415–1423. DOI : <http://dx.doi.org/10.1109/TVCG.2016.2518318>
- [31] Sandra P Marshall. 2002. The index of cognitive activity: Measuring cognitive workload. In *Proceedings of the IEEE 7th conference on Human Factors and Power Plants*. IEEE. DOI : <http://dx.doi.org/10.1109/HFPP.2002.1042860>
- [32] Silvano Martello and Paolo Toth. 1990. *Knapsack Problems: Algorithms and Computer Implementations*. John Wiley & Sons, Inc., New York, NY, USA.
- [33] Vivian Genaro Motti and Jean Vanderdonckt. 2013. A computational framework for context-aware adaptation of user interfaces. In *IEEE 7th International Conference on Research Challenges in Information Science (RCIS)*. IEEE, 1–12. DOI : <http://dx.doi.org/10.1109/RCIS.2013.6577709>
- [34] Jens Müller, Roman Rädle, and Harald Reiterer. 2016. Virtual Objects As Spatial Cues in Collaborative Mixed Reality Environments: How They Shape Communication Behavior and User Task Load. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. ACM, New York, NY, USA, 1245–1249. DOI : <http://dx.doi.org/10.1145/2858036.2858043>
- [35] Nargess Nourbakhsh, Fang Chen, Yang Wang, and Rafael A. Calvo. 2017. Detecting Users' Cognitive Load by Galvanic Skin Response with Affective Interference. *ACM Trans. Interact. Intell. Syst.* 7, 3, Article 12 (Sept. 2017), 20 pages. <http://doi.acm.org/10.1145/2960413>
- [36] Benjamin Nuernberger, Eyal Ofek, Hrvoje Benko, and Andrew D. Wilson. 2016. SnapToReality: Aligning Augmented Reality to the Real World. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. ACM, New York, NY, USA, 1233–1244. DOI : <http://dx.doi.org/10.1145/2858036.2858250>
- [37] Antti Oulasvirta. 2017. User Interface Design with Combinatorial Optimization. *Computer* 50, 1 (Jan. 2017), 40–47. DOI : <http://dx.doi.org/10.1109/MC.2017.6>

- [38] Antti Oulasvirta, Samuli De Pascale, Janin Koch, Thomas Langerak, Jussi Jokinen, Kashyap Todi, Markku Laine, Manoj Krishthombuge, Yuxi Zhu, Aliaksei Miniukovich, Gregorio Palmas, and Tino Weinkauff. 2018. Aalto Interface Metrics (AIM): A Service and Codebase for Computational GUI Evaluation. In *The 31st Annual ACM Symposium on User Interface Software and Technology Adjunct Proceedings (UIST '18 Adjunct)*. ACM, New York, NY, USA, 16–19. DOI: <http://dx.doi.org/10.1145/3266037.3266087>
- [39] Antti Oulasvirta and Andreas Karrenbauer. 2018. Combinatorial Optimization for Interface Design. In *Computational Interaction*. Oxford University Press.
- [40] Seonwook Park, Christoph Gebhardt, Roman Rädle, Anna Maria Feit, Hana Vrzakova, Niraj Ramesh Dayama, Hui-Shyong Yeo, Clemens N. Klokmoose, Aaron Quigley, Antti Oulasvirta, and Otmar Hilliges. 2018. AdaM: Adapting Multi-User Interfaces for Collaborative Environments in Real-Time. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. ACM, New York, NY, USA, Article 184, 14 pages. DOI: <http://dx.doi.org/10.1145/3173574.3173758>
- [41] Bastian Pfleging, Drea K. Fekety, Albrecht Schmidt, and Andrew L. Kun. 2016. A Model Relating Pupil Diameter to Mental Workload and Lighting Conditions. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. ACM, New York, NY, USA, 5776–5788. DOI: <http://dx.doi.org/10.1145/2858036.2858117>
- [42] Edward Rosten, Gerhard Reitmayr, and Tom Drummond. 2005. Real-Time Video Annotations for Augmented Reality. In *Proceedings of the First International Conference on Advances in Visual Computing (ISVC'05)*. Springer-Verlag, Berlin, Heidelberg, 294–302. DOI: http://dx.doi.org/10.1007/11595755_36
- [43] Sayan Sarcar, Jussi PP Jokinen, Antti Oulasvirta, Zhenxin Wang, Chaklam Silpasuwanchai, and Xiangshi Ren. 2018. Ability-Based Optimization of Touchscreen Interactions. *IEEE Pervasive Computing* 17, 1 (2018), 15–26. DOI: <http://dx.doi.org/10.1109/MPRV.2018.011591058>
- [44] John Sweller. 1988. Cognitive load during problem solving: Effects on learning. *Cognitive science* 12, 2 (1988), 257–285.
- [45] Arthur Tang, Charles Owen, Frank Biocca, and Weimin Mou. 2003. Comparative Effectiveness of Augmented Reality in Object Assembly. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '03)*. ACM, New York, NY, USA, 73–80. DOI: <http://dx.doi.org/10.1145/642611.642626>
- [46] Markus Tatzgern, Valeria Orso, Denis Kalkofen, Giulio Jacucci, Luciano Gamberini, and Dieter Schmalstieg. 2016. Adaptive information density for augmented reality displays. In *2016 IEEE Virtual Reality (IEEE VR '16)*. 83–92. DOI: <http://dx.doi.org/10.1109/VR.2016.7504691>
- [47] Beste F. Yuksel, Kurt B. Oleson, Lane Harrison, Evan M. Peck, Daniel Afergan, Remco Chang, and Robert JK Jacob. 2016. Learn Piano with BACH: An Adaptive Learning Interface That Adjusts Task Difficulty Based on Brain State. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. ACM, New York, NY, USA, 5372–5384. DOI: <http://dx.doi.org/10.1145/2858036.2858388>

APPENDIX

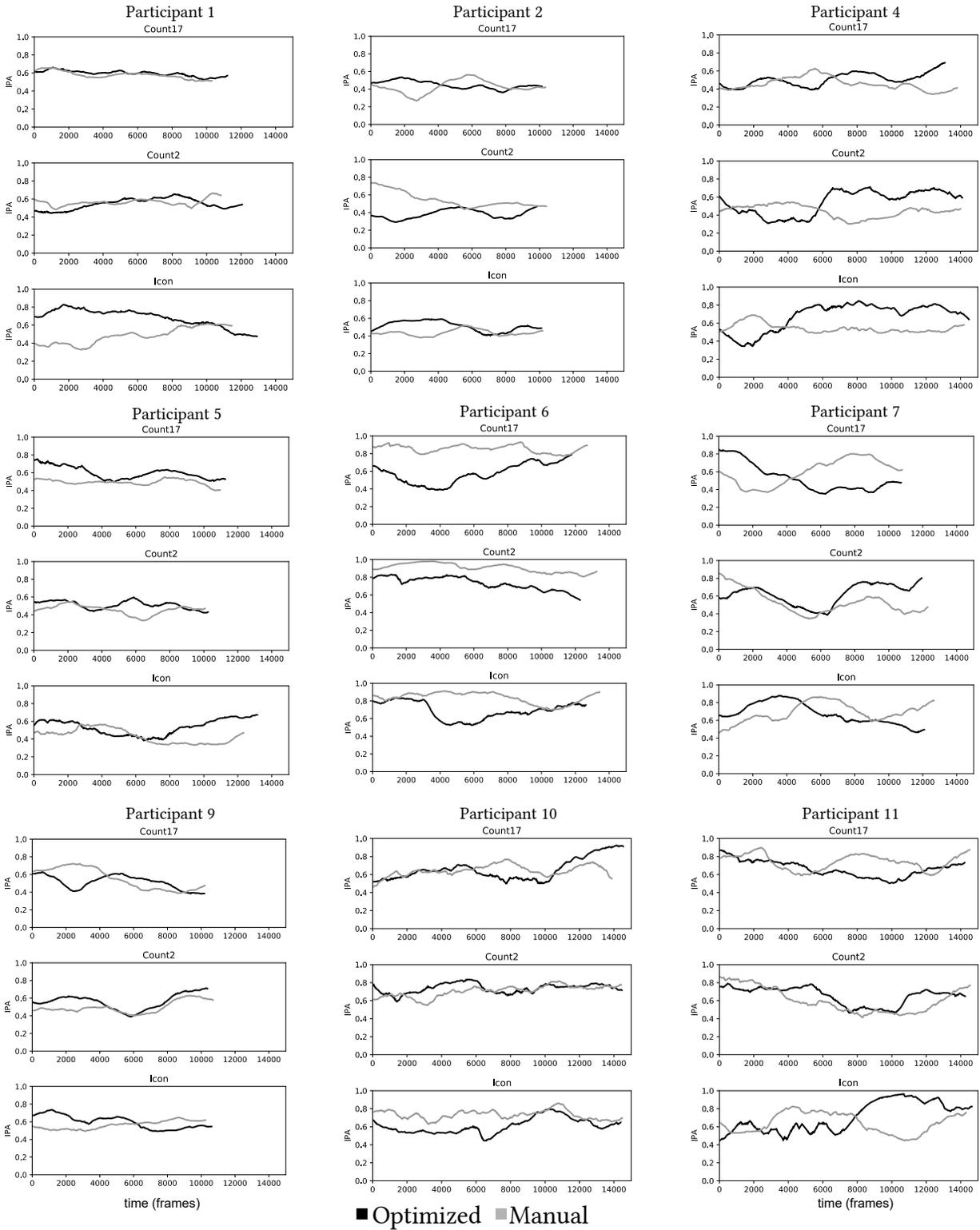


Figure 11. Computed normalized IPA values for 9 participants of our experiment per condition.