# Designing Virtual and Augmented Reality User Interfaces using Parameterized Function Structure Models

Per Ola Kristensson
University of Cambridge

**ABSTRACT**

A challenge when designing virtual and augmented reality interfaces is the fact that such interfaces can, and will, be used in a variety of contexts that cannot be anticipated or controlled by the designer. Further, since a designer cannot accurately predict how an interface will be used, and the context of use of any interface is ever changing, users will appropriate such an interface by developing their own strategies for achieving their goals. This paper introduces parameterized function structures as a tool from design engineering that may be useful in understanding and analyzing the emergent interaction properties of a virtual or augmented reality user interface at an early stage in the design process.

## 1 INTRODUCTION

One view of design is that a design is about exploring a multidimensional design space where each design dimension is a particular aspect or concern of a design, such as device size and battery length. In this model, an artefact is an instantiation in this design space—an operating point that fixes the parameters in this multidimensional design space. By moving the operating point around in this space we thus generate hypothetical artefacts. It is fair to ask the question whether we can identify an optimal operating point, however, unfortunately this is rarely the case. Consider for example a simple wearable device where we may wish to minimize device size and maximize battery life. These two design dimensions form a trade-off as by making our device smaller we inevitably limit battery life.

In general, such trade-offs between different design dimensions can be either explicit or implicit. Explicit trade-offs are trade-offs the designer is aware of and can thus make an informed decision about. Implicit trade-offs are more subtle as these are trade-offs that the designer has not explicitly considered but they are nevertheless there, emergent from the design itself. A failure to identify and properly consider such implicit trade-offs is important, however, as a failure to do so may result in a design that works well in theory but ultimately fails to be adopted by users when deployed.

However, understanding the design space and any inherent trade-offs is challenging in practice, and perhaps particularly so when designing immersive user interfaces that can be used in a variety of contexts and frequently rely on uncertain sensing to infer user intentions through noisy hand and eye tracking. Such designs are difficult to get right as there is a vast range of parameters that have to be considered by the designer. However, while there may not be an optimal operating point in this design space, there is an optimal set of trade-off decisions that balance the design dimensions that span all possible designs.

This paper presents a method from design engineering that can be adapted as a complementary tool to assist the early stage design of immersive user interfaces. The idea is to arrive at a design at a *functional* level, detailing *what* has to be done by the user interface as opposed to *how*. Once a functional architecture has been arrived at it can be parameterized, enabling the designer to generate a range of hypothetical design outcomes through simulation. In this way, parameterized function structures provide an opportunity for the designer to both diverge and converge in their design thinking: diverge by considering options for design by inspecting a range of simulated outcomes, and converge by ruling out regions of the operating space that does not result in desirable outcomes. As a complementary method, parameterized function structures cannot be the only design method, but they can guide for instance user studies in identifying promising operating points that can later be chosen as conditions in, for example, a controlled experiment.

This may be particularly important to realize seamless use of virtual and augmented reality technologies in practice. For example, recent research has shown that while it is possible to work immersed in virtual reality for an entire work week, workers struggle to cope with virtual reality technology for such an extended period of time [1]. In short, immersive interfaces may provide additional qualities to interaction but they clearly also take qualities away from how users carry out their work already. For virtual and augmented reality interfaces to achieve mainstream user adoption they must provide *additional* qualities beyond simply transplanting established interaction techniques and interaction paradigms to immersive settings. To effectively and efficiently ideate, explore, analyze, refine, and reimagine such qualities we need design methods that allow us to approach designs at an early stage of a design where we still have not committed decisions on which particular solutions we wish to implement to carry out functions.

The remainder of this paper will explain the potential role design engineering can play in designing virtual and augmented reality interfaces, introduce parameterized function structures, and discuss how this method may be able to assist a designer in exploring interactions emerging from immersive user interfaces.

## 2 DESIGN ENGINEERING

Design engineering, or engineering design [6], is a subdiscipline within engineering concerned with devising processes that allow engineers to design systems that are effective, efficient, safe, and deployable. Historically, much design research in this area was focused on mechanical and later electromechanical systems. Increasingly, there is a realization that design engineering is really about building *systems* and thus about providing methods, techniques, and toolkits that allow a designer to arrive at a system that is both built correctly, that is, a verification process succeeds in ensuring all relevant requirements have been met, and is fit for purpose, that is, that a deployed system addresses the original design problem.

Prior work in human-computer interaction has explored design engineering in the context of designing augmentative and alternative communication (AAC) interfaces for nonspeaking individuals with motor disabilities [3]. This work considered the problem that most AAC interfaces are exceedingly difficult to evaluate using traditional user-centered methodologies as AAC users are difficult to get hold of, and form a highly heterogeneous user group with diverse individual needs, wants, capabilities, and limitations. Further, adaptive AAC interfaces may require extensive exposure in order to demonstrate measurable benefits. For example, a context-aware sentence retrieval system for AAC may only be effective and efficient following months of use. However, it would be unethical to

979-8-3503-7449-0/24/$31.00 ©2024 IEEE
DOI 10.1109/VRW62533.2024.00081

subject a nonspeaking individual with motor disabilities to use a hypothetically improved system for several months without prior indication that such a system may provide an eventual tangible benefit. A solution to this issue is to adopt a design engineering method to simulate a variety of system outcomes in order to assess the viability of context-aware sentence retrieval at design time [3]. This enables the designer to not only assess whether such an approach may be effective and efficient, but also to quantify such an effect as a function of critical system parameters, such as the accuracy of the word-completion algorithm and the accuracy of the context sensing. Later work further nuanced such analyses by considering sentence generation as well as sentence retrieval, and by considering further behavioral factors, such as a user's ability to engage with an interface in an efficient manner in the face of bounded rationality, human error, and interruptions [9].

In addition to predicting future performance in situations when it is not possible to assess the viability of an interface through a traditional user study, design engineering can be used to explain *mechanisms* of interaction. One such example is word prediction [5]. For an able-bodied user, large scale empirical investigations on mobile typing has revealed that word predictions are unlikely to improve performance much [7]. It is tempting to attempt to explain such a result by carrying out an empirical user study, comparing an interface with or without word predictions. However, such a comparison is necessarily bound by the parameter choices made, such as the choice of language model, the number of word prediction slots above the keyboard, the choice of word prediction algorithm, and so on. Further, there is a *latent* set of parameters that govern whether word predictions are useful, and they encode the user's word prediction strategy [5]. For example, a user may choose to only look at word predictions if a word is of a sufficient length, the intuition being that a short word is unlikely to be correctly predicted. Alternatively, a user may choose to type $k$ keys before looking at word predictions, the intuition being that a word prediction is unlikely to be correct unless the user has inputted sufficient information. These strategies can be combined and, importantly, they cannot be directly controlled in an experiment with users as they are emergent and individual.

However, through simulation Kristensson and Müllners [5] were able to demonstrate that for an able-bodied user exhibiting a typical typing rate, the average increase in entry rate through use of word predictions is highly limited, with a possible increase in entry rate limited to approximately two words per minute, and only if the user adopted a very specific set of strategy parameters consistently. Further, implementing such an optimal strategy might be difficult for a user to achieve in practice, as even if the user is at the optimal operating point for word prediction usage, the standard deviation is high and thus in many individual cases the optimal strategy would still result in a net reduction in entry rate. In other words, word prediction performance for able-bodied users is highly reliant on the user's individual typing strategy, which is a set of latent parameters that the designer cannot directly influence [5]. This is an example of insight that would be very difficult to obtain through traditional user research methods, such as controlled experiments or think aloud studies, and where a systematic analysis of key parameters yield an explanation of not only the outcome but also the mechanism that underpins the interaction giving rise to the outcome.

## 3 Function Structures

A *function structure* is a straightforward diagrammatic technique used in design engineering to model a system at a *functional level*. By *function* we mean an abstract function that must be realized by the system to carry out its overall function. A function describes *what* a system must carry out but it does not describe *how* to do it. At a later stage in a design, functions must be translated to *function carriers*, solutions, that describe a concrete implementations of functions. For example, a wearable device may have the function `Supply Energy`

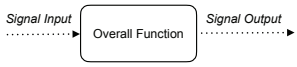


Figure 1: A function structure denoting a generic `Overall Function` that receives a signal and emits a signal.

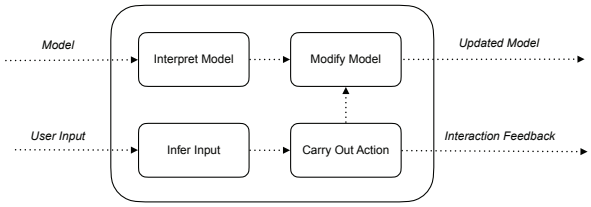


Figure 2: An example generic function structure model for manipulation of a model, such as a 3D model, that receives a model and user input as input signals and emits an updated model and user feedback as output signals.

and this function may be addressed by a function carrier, such as a 5V DC battery.

Function structures are evolved by first considering an overall function (Figure 1). This overall function is a high-level function, such as, for example, `Create 3D Model`. This overall function receives *signals*, such as, *Pointing Action* or *Selection Action*. As a consequence, the overall function will emit signals, such as user feedback and, for example, change signals to modify the structure or properties in a 3D model.

This overall function can subsequently be decomposed into subfunctions that further elaborate on the necessary functions to carry out the overall function. Such an elaboration results in a flow of signals across functions that provide insight into the structure of the system.

Figure 2 shows an example of a function structure for a hypothetical overall function `Change Model` that allows a user to modify some aspect of a model, such as a 3D model. Such manipulation requires the overall function to receive some form of user input, which is encoded as a *User Input* signal. This signal, which may be, for example, a hand or eye tracking signal, is decoded by the system using an `Infer Input` function and can subsequently be executed by a `Carry Out Action` function. This latter function will send a signal to the function `Modify Model`. The second input signal to the overall function is *Model*, which is some representation of the model of interest. This model is interpreted by the system using `Interpret Model`, which translates it into a representation that can be subsequently be manipulated by `Manipulate Model`. Ultimately, the overall function emits two signals in response to a model and some unspecified user input: *Updated Model* and *Interaction Feedback*.

As we are designing at the *functional* level, as opposed to the function carrier level, it is important the functional architecture is designed to be as *solution-neutral* as possible. This means we need to ensure we are at an abstraction level that simultaneously allows us to meaningfully articulate the key functions that are essential to carry out the overall function while at the same time avoid design fixation by prematurely limiting the exploration to narrowly defined predetermined solutions.

In practice, it may be challenging to set the abstraction at the right level and there is no one answer. Fundamentally, the abstraction level will depend on the nature of the problem. For example, if the system design problem is tackling eye gaze input it is justifiable to define input signals to refer directly to eye gaze. Further, if the sys-

tem design problem is specifically about the semantics of selecting several visual elements through gaze, as opposed to detecting gaze to begin with, it may be further justifiable to define input signals in terms of fixations and saccades.

Related, actual systems consist of a vast number of functions and a diagrammatic approach can easily be overwhelmed if there is an attempt to capture *all* interaction aspects. As a complementary technique, a function structure model is most useful when modeling specific aspects of interaction, such as a specific selection interaction technique or object manipulation step that can be subsequently parameterized and analyzed.

## 4 Parameters and Emergent Outcomes

Having arrived at a function model of some aspect of interaction we are interested in, we can now parameterize the model by inspecting the functions and signals and assign them relevant parameters. Fundamentally there are two classes of parameters:

**Controllable parameters** These are parameters that are under the influence of the designer. Thus, knowledge of such parameters allow the designer to tune or optimize these parameters to maximize some desirable outcome. Examples of such parameters are programmed timeout durations, the number and appearance of visual elements, and so on.

**Uncontrollable parameters** These are parameters that cannot be set at design time. Such parameters may be intrinsic to the individual user, such as response time, and input sensing, such as accuracy. They also include latent parameters, such as a user's specific strategy to achieve their goal. In general, an understanding of uncontrollable parameters allows sensitivity analysis—computationally investigating how sensitive system outcomes are to fluctuations in parameter values. In addition, it allows an exploration into mechanisms that may lead to certain outcomes. For example, it may be possible to investigate how system outcomes may vary as a result of strategy parameters encoding a hypothetical user's multiple choices in carrying out a task. This may suggest different ways of supporting users in carrying out tasks.

In general, knowledge of controllable and uncontrollable parameter values and ranges that result in desirable system outcomes can be used to improve the requirements specification of an immersive user interface. This may be particularly useful at an early stage in a design, to understand the the optimal controllable parameters settings that are the most likely to result in robust and desirable system outcomes, such as a consistent accuracy that is above some tolerance level. In addition, it is possible to gather requirements. For example, if there is a machine learning component classifying user input into distinct actions, it is useful to understand what average accuracy is required to yield satisfactory outcomes on average. Even though such a parameter is ultimately uncontrollable, as we cannot usually prescribe user input, we can assess its influence of performance.

Having identified the key parameters we are interested in, we can investigate emergent outcomes through envelope analysis [5]. As long as we are careful to limit our analysis to just the key functions, signals, and interaction-critical parameters we can simulate emergent outcomes by analyzing how they interact to give rise to emergent system outcomes. Such an analysis is specific to the type of interaction but can in many cases be designed using a simple set of equations that can be implemented in just a few lines of code.

Figure 3 illustrates such a hypothetical envelope analysis as a function of two parameters $x$ and $y$. The dashed curve indicates the ridge between acceptable and unacceptable performance. The interaction of the two parameters that give rise to the emergent outcome is visible in the plot. Creating a set of such plots allow a visualization of the possible operating points of the design for a particular desired emergent outcome.

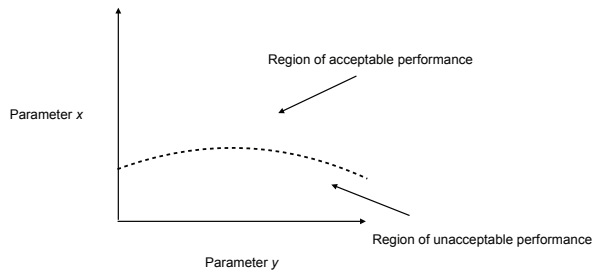


Figure 3: A hypothetical envelope analysis of two design parameters $x$ and $y$. The $z$ coordinate in the plot is indicating an emergent quality or performance metric. The dashed curve indicates the ridge that separates the regions of acceptable and unacceptable performance. The influence of the design parameters can be understood through their interrelated roles on the emergent outcomes in this plot.

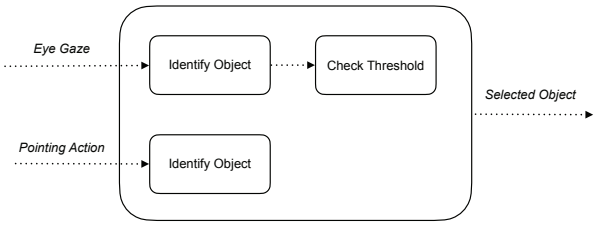


Figure 4: A function structure modeling the high-level action of a user selecting a distant object in an augmented reality interface by simultaneously gazing at the object and pointing towards it with their hand. The system needs to identify the object using two separate functions. In addition, the user must maintain their eye gaze fixation on the object of interest for a set duration threshold, managed by `Check Threshold`. The details of how the *Eye Gaze* and *Pointing Action* signals are interpreted and ultimately fused is omitted in the model.

## 5 Example

Consider a hypothetical augmented reality interface that contains a large set of distant virtual objects that the user desires to select. Now assume a designer is interested in exploring a multimodal target acquisition technique in which the user uses their gaze to fixate at the distant virtual object of interest for a set time duration $T$, and simultaneously points in the direction of the target with some accuracy $A \in [0,1]$, where 0 means no accuracy and 1 means complete accuracy. Figure 4 illustrates a high-level function structure that captures this interaction technique.

If we assume eye gaze fixation detection is reasonably accurate, the duration $T$ required of the user to fixate on the desired key is a controllable parameter that the designer can set. Its setting is a trade-off: a too low setting will trigger false object selections, while a too high setting will needlessly slow down object selection time. Let us model the probability of success for the user to select a distant target as a combination of a measure of the user's accuracy in successfully pointing to the distant target, and the probability $p$ of the user successfully maintaining their fixation on the distant target using this simplified model: $p = 1 - e^{-\lambda T}$, where $T$ is the timeout threshold and $\lambda$ is a model parameter, here set to 0.5.

Figure 5 shows a resulting heatmap that visualizes the estimated probability of successful object selection using the heatmap color scale. This estimation is a function of two parameters: (1) the uncontrollable parameter, which is the user's accuracy in pointing at

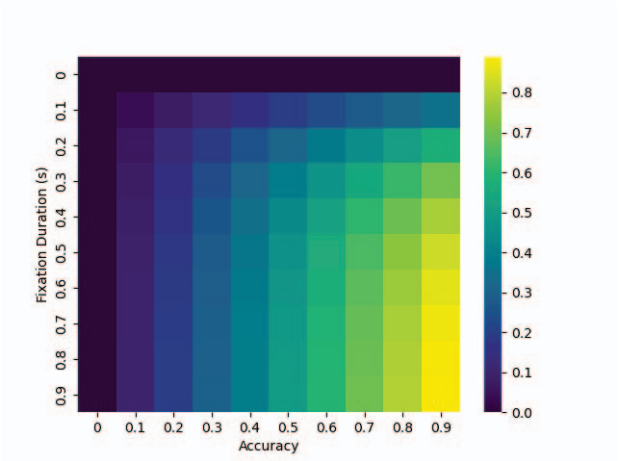


Figure 5: A heatmap visualizing the emergent operating points of a hypothetical object selection technique for an augmented reality interface combining a user's eye fixation on the object with a user pointing in the direction of the object. The heatmap shows the estimated probability of success as a function of one controllable parameter, the fixation duration threshold, and one uncontrollable parameter, the accuracy a user exhibits when pointing towards an object. The color of the heatmap is the probability of success, which is an emergent quality of the system, here assumed to solely dependent on the interaction of these two parameters. We observe that accuracy is affected by the fixation duration threshold set by the designer: a too low threshold yields a very low probability of success. On the other hand, if users are able to exhibit high accuracy in pointing towards an object, the fixation duration threshold need not be set too high. For example, the standard fixation duration threshold of 1 s is unnecessarily high according to this model, as long as user's exhibit an average accuracy that is higher than approximately 80%.

the object of interest; and (2) the controllable parameter, which is the fixation duration threshold set by the designer in seconds. While this is a highly simplified model we can already observe emergent properties of this system.

First, the fixation duration threshold is critical as certain regions result in a very low predicted probability of success. Second, when the user exhibits a certain level of accuracy in pointing towards an object, approximately higher than 80%, the fixation duration threshold does not need to be set excessively high according to the model.

This analysis suggests a desirable requirement on the user's ability to accurately point towards a desired object. Obviously, while 100% accuracy is always desired, in practice it is sufficient to obtain an accuracy of approximately 80% as long as the fixation duration threshold is set sufficiently high. Further, we have gained some initial understanding on how to tune, or optimize, the fixation duration threshold in our system. We are now in a position to either further elaborate on the model by incorporating further parameters, carrying out sensitivity analysis, and so on, or carrying out further investigations, such as user studies, to validate that our model assumptions are sound.

## 6 Conclusion and Outlook

This paper has introduced parameterized function structures as a complementary design method for virtual and augmented reality interfaces. This method allows an aspect of interaction of interest to be closely modeled at the functional level, without assigning it a specific function carrier. Such function structure models can then be parameterized into interrelated essential controllable and uncontrollable parameters. This allows the simulation of emergent outcomes of the interaction of interest and explanations of mechanisms yielding particular outcomes. Such information is particularly valuable at the early stage of the design process of virtual and augmented reality interfaces, which give rise to use contexts and user strategies that may be exceedingly difficult to anticipate or model at design time otherwise.

It is not an entirely novel approach, simulation has been used before in human-computer interaction to evaluate different ways of allowing users to achieve their goals (e.g. KLM-GOMS [2]), and there is prior work (e.g. [8]) that carefully assesses emergent design parameters through simulation, sometimes guided by findings in qualitative research work (e.g. [4]). However, there is value in formalizing such computational explorations in terms of designing systems at the functional level and assessing emergent system qualities through computational simulations of parameters that are as solution-neutral as possible. It allows early insights into possible designs, guides more elaborate follow-up studies, such as empirical investigations, and may allow analysis of interaction that cannot be otherwise carried out, such as exploring system outcomes as a function of user strategy.

In terms of the outlook on this approach to design virtual and augmented reality interfaces, there are several opportunities, challenges, and research trajectories:

**Case studies** Which particular interaction scenarios in virtual and augmented reality lends itself to this type of analysis? What are particular aspects of systems that can be abstracted and isolated and made tractable to allow a fruitful application of this approach?

**Frameworks and toolkits** Can we improve outcomes when designing virtual and augmented reality interfaces by establishing frameworks that help a designer to quickly design their system at the functional level? Can we create toolkits that enable the designer to easily encode a variety of parameter assumptions underpinning their designed interaction, thereby allowing them to easily analyze emergent outcomes and extract requirements on the design?

**Explaining observed behavior** As alluded to earlier, parameterized function structures have previously been used to illuminate the mechanisms explaining why word prediction overall does not benefit able-bodied users typing on mobile keyboards. Can we use this approach to reveal mechanisms of interaction that can explain results in virtual and augmented reality interfaces?

**Knowing what to look for** A fundamental problem in user research is *knowing what to look for*. It is easy to be led astray and focus on elements that simply happen to be easy to gather evidence for. Can we use parameterized function structures and envelope analysis to generate questions about a design that can better guide the researcher towards investigations focusing on key parameter values, or interactions between parameters? For instance, can we create more informative experimental designs where the conditions are informed by parameter choices suggested by envelope analysis?

## Acknowledgments

This work was supported by EPSRC (grant EP/W02456X/1).

## References

[1] V. Biener, S. Kalamkar, N. Nouri, E. Ofek, M. Pahud, J. J. Dudley, J. Hu, P. O. Kristensson, M. Weerasinghe, K. Č. Pucihar, et al. Quantifying the effects of working in vr for one week. *IEEE Transactions on Visualization and Computer Graphics*, 28(11):3810–3820, 2022.

[2] S. Card, T. Moran, and A. Newell. The Psychology of Human-Computer Interaction. 1983.

[3] P. O. Kristensson, J. Lilley, R. Black, and A. Waller. A design engineering approach for quantitatively exploring context-aware sentence retrieval for nonspeaking individuals with motor disabilities. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, pp. 1–11, 2020.

[4] P. O. Kristensson, M. Mjelde, and K. Vertanen. Understanding adoption barriers to dwell-free eye-typing: Design implications from a qualitative deployment study and computational simulations. In *Proceedings of the 28th International Conference on Intelligent User Interfaces*, pp. 607–620, 2023.

[5] P. O. Kristensson and T. Müllners. Design and analysis of intelligent text entry systems with function structure models and envelope analysis. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, pp. 1–12, 2021.

[6] G. Pahl and W. Beitz. *Engineering Design: A Systematic Approach*. Springer Science & Business Media, 2013.

[7] K. Palin, A. M. Feit, S. Kim, P. O. Kristensson, and A. Oulasvirta. How do people type on mobile devices? observations from a study with 37,000 volunteers. In *Proceedings of the 21st International Conference on Human-Computer Interaction with Mobile Devices and Services*, pp. 1–12, 2019.

[8] K. Vertanen, D. Gaines, C. Fletcher, A. M. Stanage, R. Watling, and P. O. Kristensson. Velociwatch: Designing and evaluating a virtual keyboard for the input of challenging text. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, pp. 1–14, 2019.

[9] B. Yang and P. O. Kristensson. Imperfect surrogate users: Understanding performance implications of augmentative and alternative communication systems through bounded rationality, human error, and interruption modeling. *Proceedings of the ACM on Human-Computer Interaction*, 7(MHCI):1–33, 2023.