BlocklyVR: Exploring Block-based Programming in Virtual Reality

Martin Hedlund
marthed@kth.se
KTH Royal Institute of Technology
Stockholm, Sweden

Adam Jonsson
adajon@kth.se
KTH Royal Institute of Technology
Stockholm, Sweden

Cristian Bogdan
cristi@kth.se
KTH Royal Institute of Technology
Stockholm, Sweden

Gerrit Meixner
gerrit.meixner@hs-heilbronn.de
Heilbronn University
Heilbronn, Germany

Elin Ekblom Bak
elin.ekblomBak@gih.se
GIH - The Swedish School of Sports and Health Sciences
Stockholm, Sweden

Andrii Matviienko
andriim@kth.se
KTH Royal Institute of Technology
Stockholm, Sweden

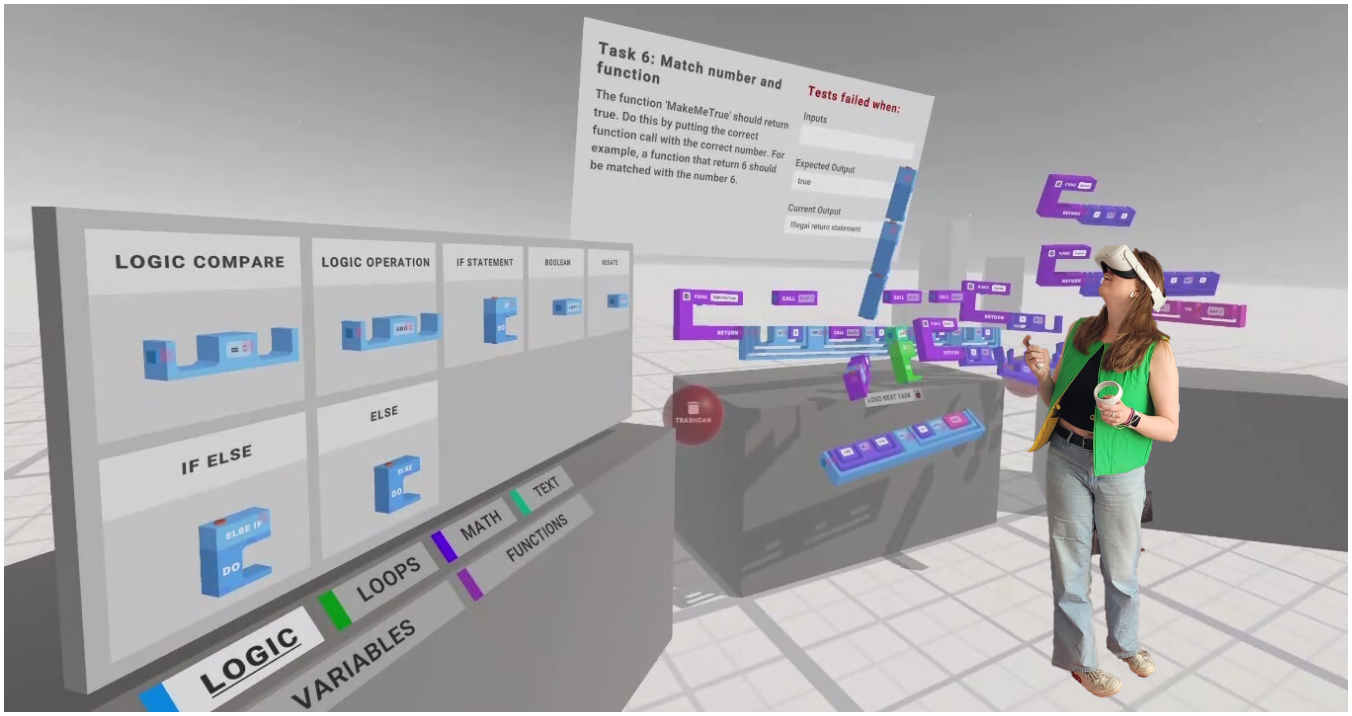

Figure 1: BlocklyVR is Virtual Reality programming environment that allows users to be within the scene and interact with virtual blocks. The interaction is enabled via a VR headset connected to two controllers in a room-size physical space.

ABSTRACT

As programming is typically a static activity in front of a screen, we perform an initial exploration around the capabilities of block-based programming in the immersive space using Virtual Reality (VR) to make an early charting on how programming could involve moving the programmer's body. We created a block-based programming interface in a VR space called BlocklyVR based on the existing Blockly programming environment. To investigate programmer performance and experience in BlocklyVR, we conducted a controlled lab experiment (N = 20) with eight programming tasks that covered mathematical operations, if-statements, and function creation. Our initial exploration contributes by classifying movement types made by BlocklyVR programmers and reflecting on how these movements are related to the programming tasks. Additionally, our data suggests that participant performance in BlocklyVR was not affected compared to the 2D Blockly, even if participants were physically moving in VR space. We also found that the virtual

MUM '23, December 03–06, 2023, Vienna, Austria

ACM ISBN 979-8-4007-0921-0/23/12.
https://doi.org/10.1145/3626705.3627779

reality sickness was marginal. Lastly, we identified four types of interaction that can potentially be employed by VR designers and developers aiming to convert a static task, like programming at a desk, into a "mobile" immersive experience.

CCS CONCEPTS

• **Human-centered computing → Virtual reality**; **User studies**; **Empirical studies in HCI**.

KEYWORDS

virtual reality, programming, blockly, physical movement

ACM Reference Format:
Martin Hedlund, Adam Jonsson, Cristian Bogdan, Gerrit Meixner, Elin Ekblom Bak, and Andrii Matviienko. 2023. BlocklyVR: Exploring Block-based Programming in Virtual Reality. In *International Conference on Mobile and Ubiquitous Multimedia (MUM '23), December 03–06, 2023, Vienna, Austria.* ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3626705.3627779

1 INTRODUCTION

Researchers in Human-Computer Interaction have a long-going interest in creating engaging interfaces for programming [15, 55, 63]. Two forms of programming interfaces became particularly interesting to explore due to their playfulness and suitability for non-tech-savvy users: block-based and tangible. For this, researchers proposed and developed numerous block-based and tangible programming interfaces that made programming more accessible to non-programmers [4, 5, 29, 47, 50, 65, 74, 75]. Block-based interfaces conceptually focus on making programming easier by presenting programming concepts as visually distinct blocks. However, the traditional method of engaging with the blocks on a computer screen using a mouse or touch interface hinders the user's ability to fully immerse themselves in a physically engaging way rather than an external party behind the glass screen [52]. Tangible programming interfaces bring the interaction away from the computer screen but are limited by the physical materials representing the program constructs [68]. Virtual Reality (VR) provides an interesting opportunity to create engaging programming interfaces without these constraints. Several projects have begun to explore the combination of VR and block-based interfaces [60, 64, 67, 72]. However, there still needs to be a greater understanding of what advantages the Virtual Reality space can introduce for block-based programming.

Previously, researchers have employed block-based programming languages in educational settings for learning programming [43, 53, 58], and as an easy-to-use approach for constructing simple software for end-users, e.g., hobbyist electronic projects, mobile apps, and games [5]. Likewise, tangible programming interfaces have been used for similar purposes, such as fostering computational thinking [24, 46, 70], creating 3D models [35], and supporting visually impaired people [25]. Tangible interfaces can foster physical sensemaking and engagement [7, 22, 48], but are limited by the physical constraints imposed by the materials. VR can provide an alternative to tangible interaction by facilitating embodied physical interaction since users can freely move while assembling blocks into programs. Previous block-based VR interfaces have been engaging for both novice students [72] and K-12 children [60], and direct object interaction instead of pointer-based interfaces might be a contributing factor. Incorporating both hands in the interaction [72] and standing up instead of sitting [64] could be additional advantages, but there is a lack of empirical evaluation confirming these assumptions.

In this paper, we explore the possibilities and limitations of block-based programming in Virtual Reality spaces with the long-term aim of enriching the desk-bound programmer's work. For this, we created an early exploratory prototype for interactive programming in Virtual Reality called BlocklyVR, based on the block-based interface Blockly [1]. With this, we aim to facilitate spatial, direct, and two-handed interaction with programming blocks. To evaluate the effectiveness of BlocklyVR regarding the extra time the programmer might spend due to movement in space, we conducted a controlled lab experiment (N = 20) comparing our proposed system to the existing 2D Blockly programming environment. Additionally, for the BlocklyVR participants (N = 10), we measured physical activity and conducted video observations to identify types of movements in a physical programming environment. We found that task completion time is comparable between Blockly and BlocklyVR, except for one task characterized by low complexity and unproductive movement. We also classified four distinct types of interaction and movement in BlocklyVR. Our findings suggest implications for efficiently incorporating physical activity in productive VR interfaces, which sets a future research direction for designing Virtual Reality systems.

2 RELATED WORK

Although few empirical evaluations have focused on exploring block-based programming in Virtual Reality (VR) and its influence on user performance and physical movement, researchers have created and investigated many systems to support block-based, tangible, and VR programming. In this section, we outline these two pillars of the previous work we build on in this paper: (1) block-based & tangible programming and (2) programming in VR.

2.1 Block-based & Tangible programming

Block-based programming grew from the MIT MediaLab on LogoBlocks research in the 1980s [47]. Seymour Papert, one of the principal creators of the Logo programming language used in LogoBlocks, stated that he intended to create an "immersive environment" for learning mathematical concepts [49]. It has gained significant popularity in the last decade due to the rise of platforms like Alice [5], Scratch [47], Snap! [29], and Blockly [4, 75]. These platforms offer versatile interactive environments for novice users to explore programming for robotics [61, 75], tangibles [39], IoT [4, 59, 74], and the LEGO Robotics ecosystem [65].

Block-based programming is conceptually based on making programming easier and more accessible. Drawing on the principles behind the move from text-based to graphical user interfaces in the 1980s, it relies on direct manipulation, with a main principle of having the "knowledge" - in this case, the programming syntax - available in the "world" (on the screen), rather than in the "head" [62]. Apart from the direct manipulation mode of finding and arranging commands, the blocks draw on principles of mapping, grouping, and visual affordance, with the shapes and colors of

[1] https://developers.google.com/blockly

the blocks indicating their type and affordances. However, today's block-based programming experience primarily focuses on the 2D space, where a user sits in front of a screen and indirectly manipulates the blocks using a mouse and keyboard or touch input on a smartphone or tablet [2]. In our work, we go beyond the interaction "behind the glass" and explore the possibilities of embodied physical engagement with blocks for programming as we explore the design space of "raising programmers from the desk".

One possibility of incorporating physical interaction for programming is through tangible interaction. Along with the development of block-based interfaces, researchers employed tangible interaction for learning to program for over 30 years [34, 51] and discovered that it could foster enhanced physical sensemaking and engagement [7, 22, 46, 48]. Using Virtual Reality sacrifices some of the richness of touching and sensing physical material but could still provide an embodied experience compared to a sedentary desktop setting [52]. Some affordances of a tangible interface can also be mimicked in VR. For example, audio and haptic feedback can simulate weight sensations [73]. More importantly, since all material is virtual in VR, it facilitates dynamic modifications central to any programming activity, e.g., interface actions such as making copies, changing the behavior of objects, and changing shape and size related to object manipulations, which we build on in our work.

2.2 Programming in Virtual Reality

Programming in Virtual Reality covers a plethora of programming activities, such as live programming, virtual scene creation, code comprehension, and learning. Live programming [23] provides immediate feedback on output changes and the benefits of immersion in the virtual space. For example, Castelo-Branco et al. [10] studied live programming in VR for architectural 3D models and received positive feedback from end-users. However, users found using a physical keyboard cumbersome and suggested direct manipulation techniques for code interaction, which we employed in our work. Another research direction explored the creation of virtual scenes with virtual controls. For instance, EngtangleVR [11] used a hybrid approach in which certain details of VR scenes can be edited with sliders and checkboxes within the virtual environment and the rest of the scene – with a visual-programming interface. The example of FlowMatic [77] put a visual-programming interface inside the VR scene. Users can thereby directly program the behavior of objects in the scene without taking off their VR headsets. Some projects have employed VR for program comprehension [19] that entails tracing a program flow [12] by visualizing it on a 3D spatial layout. For example, Dominic et al. [16] compared programmers' comprehension of Java code in VR with a desktop setting and found that their implementation made comprehension more difficult. In contrast, a study on ExplorViz [31], a tool for 3D-visualizing Java code in VR, demonstrated that participants solved more comprehensive tasks correctly in ExplorViz compared to browsing the textual code base, but zooming in and out of the 3D-visualization was found cumbersome. Lastly, research projects focused on learning programming in immersive spaces [37, 64, 67, 71]. For example, Cubely [72] and VR-OCKS [60] are two block-based environments that allow the users to walk around and directly interact with the blocks instead of sitting down and interacting with a ray-cast pointer. Participants preferred Cubely over the desktop interface Blockly due to its immersiveness and two-handed interaction [72].

While Cubely and VR-OCKS are similar to our system, our questions are different, and the data we gather for exploratory purposes enables us to draw specific early lessons on transferring a desk-bound activity to VR. We introduce BlocklyVR, a Virtual Reality version of 2D-based Blockly for desktops, as an initial step in exploring non-desk-bound programming. To assess its effectiveness, we compare the performance with Blockly. Since we designed our interface to use physical space for interaction, we measured users' physical activity and related that to performance and activities. The following sections detail design considerations, our proposed system BlocklyVR, and evaluation.

3 BLOCKLYVR

BlocklyVR facilitates programming in Virtual Reality through immersive interaction with virtual blocks. The interaction is enabled via a VR headset connected to two controllers in a room-size physical space. In the following four subsections, we describe the design considerations for BlocklyVR, the programming environment, interaction concept, and implementation.

3.1 Design considerations

While the basic structure and appearance of BlocklyVR are similar to the desktop-based Blockly interface, we have introduced several design considerations necessary for block interaction in a VR space.

3.1.1 Spatial working area. The first design consideration concerns physical space for walking and interaction in a 3D environment. For example, to drag and drop blocks, a user can walk near a block and move it to a new location. A spatial work area should allow users to position blocks nearby and at any height to iterate different solutions quickly in 3D space avoiding arm fatigue and discomfort [21, 36].

3.1.2 Direct and two-handed interaction. Using a pointer in 3D space for selection is common but can be frustrating [64] and time-consuming [67]. Therefore, the second design consideration concerns walking to the blocks for direct manipulation. Directly grabbing the blocks, e.g., using controllers, mimics a tangible programming experience, which could support users' sense-making [22, 48] and engagement [60, 72]. While block-based programming in 2D involves interaction with a mouse with one hand, 6DOF VR controllers and 3D space facilitate the use of both hands for the manipulation of blocks. For example, users can interact with blocks by holding them in one hand and manipulating them with the other, as typically done in everyday interaction [26].

3.1.3 Block design. The third design consideration concerns visual cues indicating a possibility of connecting the blocks. Compared to 2D space, in 3D space, there is a need to design cues indicating which blocks can be connected, especially when viewed from different angles. As in Blockly, blocks have "connectors", i.e., zones where they can attach to other blocks.

3.1.4 Text input. Despite the advantages of block-based programming in VR, the text input still needs to be improved since users

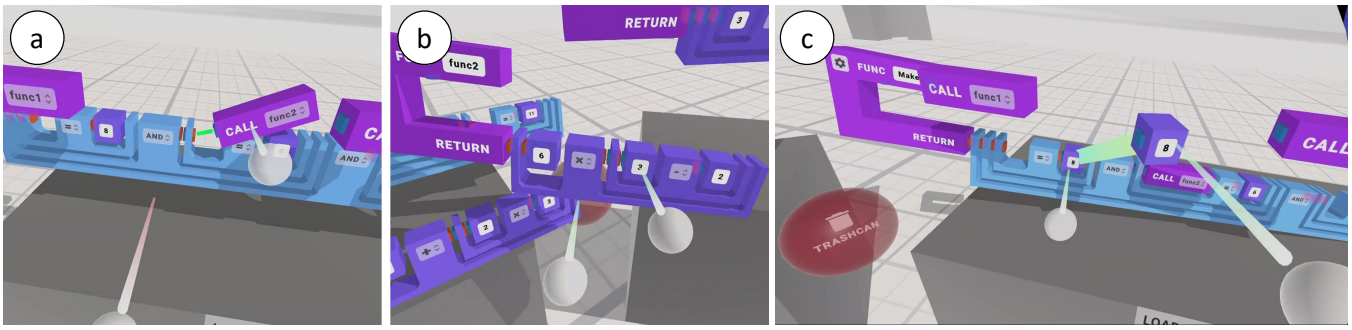

Figure 2: BlocklyVR facilitates three main interaction methods: (a) connecting, (b) disconnecting and (c) duplicating blocks.

do not see a physical keyboard. Block-based programming requires less text input than textual programming but is still necessary to declare variables and functions or to set values. Therefore, a virtual keyboard is the fourth design consideration for the text input to take users' input from pointing at virtual keys. This has been shown to be more efficient than alternative text-input solutions for VR [66].

3.2 Programming environment

In BlocklyVR, users can create programs by assembling and modifying blocks using the same logic as in Blockly. The BlocklyVR environment consists of a workspace area, a menu of programming blocks, a task instruction screen with input and output, three tables that function as boundary cues, and two trash cans (Figure 3). The workspace occupies around 6 x 2m of physical space. To delete blocks, a user can walk up to the trash cans and put them inside. A menu for selecting blocks is placed on the left side of the environment and has the following categories: logic, loops, math, text, lists, variables, and functions. Additionally, we increased the size of the blocks compared to their 2D counterparts, e.g., a while-loop block is 2.5cm in Blockly and 17cm in BlocklyVR, to ease selection in mid-air with controllers and increase text readability on the blocks due to the VR-headset resolution.

BlocklyVR features a set of programming tasks that users can complete. Each task has a test suite that checks if the assembled block code is correctly given different combinations of input values. A task screen with instructions is placed in the center where participants can assess their work progress. The task screen also updates and displays the current test suite's input, output, and expected output. If all tests have passed, the display changes, a sound is played, and the user can advance to the next task. A button underneath the table is used for proceeding to the next task.

3.3 Interaction Concept

From the interaction perspective, BlocklyVR facilitates three main interaction methods: (1) connecting, (2) disconnecting, and (3) duplicating blocks. To aid precision selection, the controllers are visually represented as a "ball and chopstick," i.e., a short ray cast.

3.3.1 Connection/Disconnection. To connect the blocks, a user must first bring them close to each other. This is done by selecting a block nearby by holding the trigger button on the controller and

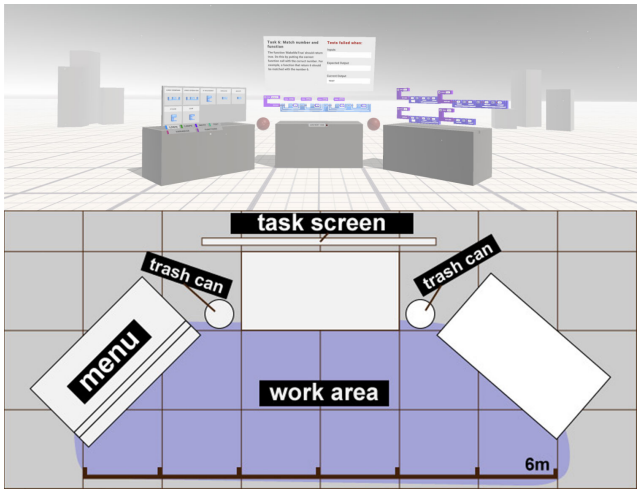

Figure 3: The BlocklyVR consists of: (1) a work area (the walkable workspace area is approximately 6x2m) with three tables acting as boundary cues for the area, 2) a menu with different block categories (logic, loops, math, text, lists, variables and functions), 3) a task screen with task instructions and test suite input and output for users to self-assess their work, 4) two trash cans for discarding blocks.

repositioning the block within range of another block. When two blocks' "connectors" (connection areas) are within range (5cm), a green line appears, indicating they can be connected (Figure 2 a). In the presence of a green line, a user can release the trigger button on the controller to connect the blocks. Disconnecting blocks requires the user to use both hands. For this, users hold a block structure (two or more connected blocks) with one hand and use the other hand to select a block they would like to disconnect with a trigger button and move it outside of the connecting range while holding a trigger button (Figure 2 b). We made the blocks slightly transparent so the connectors would be visible from all angles.

3.3.2 Duplication. To duplicate a block, a user has to grab it with both hands and "drag out" a copy of it. When the drag starts, a line appears, and to complete the duplication, the line has to be stretched enough until it snaps (Figure 2 c). Audio and haptic feedback is used to indicate how far the line is from snapping.

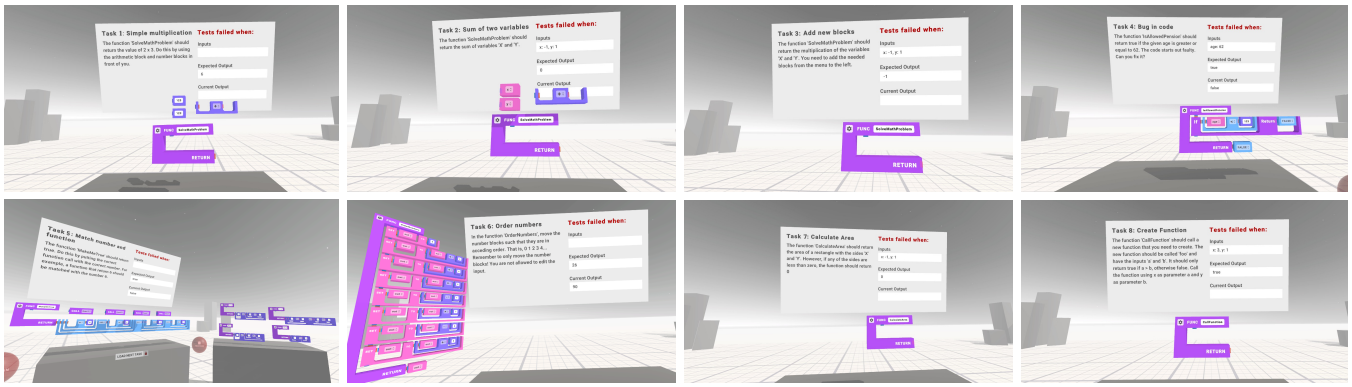

Figure 4: Visual overview of all tasks in BlocklyVR. The tasks included summation with and without menu selection, debugging, matching numbers and functions, ordering numbers, calculating area, and creating functions.

3.4 Implementation

BlocklyVR consists of the VR application created in Unity and a Node.js web server [2]. When blocks are added or modified in the VR environment, the resulting code is translated into the JavaScript version of Blockly and sent over HTTP to the server. The server executes the JavaScript code, runs a test suite for a given task, and returns the results of the tests. The server also hosts a web interface built in React for 2D Blockly. This interface includes a task instructions panel we used in the user study.

4 EVALUATION

To explore the impact of BlocklyVR on users' programming performance, we conducted a controlled lab experiment comparing it to the regular 2D Blockly on a desktop. To further understand the impact of BlocklyVR on participants' interactions and task performance, we also measured participants' physical activity and compared it with recorded video material. Lastly, interviews with BlocklyVR participants were conducted to provide qualitative feedback. The research question for this experiment is: *"How does VR block-based programming influence users' programming performance, physical activity, and experience?"*

4.1 Participants

We recruited 20 participants (15 male, 5 female) aged between 21 and 41 ($M = 26.2, SD = 5.7$) with no block-based programming experience and limited prior experience of virtual reality (VR). The participants were recruited through the advertising channels of our institution. Participants received 15€ cinema voucher as compensation for their participation.

4.2 Study design

For programming performance, the study was designed to be between-subject with one independent variable: *programming environment*. The first type of programming environment included Blockly [3] as a baseline for existing programming practices on the 2D screen. The second one is our proposed BlocklyVR programming environment as described in the previous section. The primary objective for comparing BlocklyVR to Blockly is two-fold: (1) Blockly is a state-of-the-art block-based programming environment that facilitates programming without writing a code and, thus, comparing it to its VR version will help us better understand if VR can improve users' performance and experience due to its immersive nature and, and (2) By comparing a desktop activity (in this case programming) to the same activity in VR, we aim to understand how to design future VR workspaces and how physical movement as an integral part of solving a task affects user performance and experience. We evaluated both programming environments in the between-subject study to decrease the influence of fatigue since participants had to finish eight programming tasks within 30 minutes and to avoid learning effects since we aimed to provide participants with the same set of tasks for comparability reasons. These tasks included summation with and without menu selection, debugging, matching numbers and functions, ordering numbers, calculating area, and creating a function (see Table 1 for textual and Figure 4 for a visual overview of the tasks). The tasks were always completed in the same order (from 1 to 8). We designed each task to increase in difficulty from the previous. The control group using Blockly consisted of 7 male and 3 female participants aged between 22 and 41 ($M = 24.4, SD = 5.2$), and the second group using BlocklyVR consisted of 8 male and 2 female participants aged between 21 and 38 ($M = 28.1, SD = 5.7$).

4.3 Apparatus

For the Blockly setup, we used a 16-inch Macbook Pro with a screen resolution of 2560 × 1600 and a mouse. For the BlocklyVR setup, we employed Oculus Quest 2 with both controllers and an accelerometer placed on the right hip to measure physical activity.

4.4 Measurements

We measured the following variables:

- **Task completion time (in sec)**: for each task, we measured the time it took participants to finish it. The timer started each time a "load next task" button was pressed and

[2] https://github.com/(authorname)/CodeVR
[3] https://developers.google.com/blockly

Task	Description
1: Multiplication	The function should return the value of 2 x 3.
2: Summation without the menu	The function should return a sum of two variables.
3: Summation with the menu	The function should return the sum of two variables, but in this case, users had to find the corresponding blocks in the menu.
4: Debug	The function should return true if the given age is ≥ 62.
5: Match number and function	Connect numbers with functions that return the number. For example, a function that returns six should be matched with the number 6.
6: Order numbers	Move the number blocks in ascending order.
7: Calculate area	The function should return the area of a rectangle, and if any of the sides are less than zero, it should return 0.
8: Create function	The function should call a new function you must create. The new function should be called 'foo' with the inputs 'a' and 'b' and return return true if a > b and be false otherwise.

Table 1: The overview of tasks that participants had to solve in the experiment for both programming environments.

stopped when the task was successfully completed. During the experiment, all tasks were completed successfully.

- **Physical activity duration (in %)**: for each task in BlocklyVR, we measured the percentage of time performing physical activities using acceleration along all three axes using an ActiGraph GT3-X accelerometer strapped to VR participants' right hip [57]. Acceleration data is commonly labeled according to established cut-off points that correspond to *sedentary*, e.g., standing still, *low physical movement*, e.g., light walk or slow movements, and *moderate* physical activity, e.g., regular walk or movements with higher acceleration [57]. We used these cut-off points to determine the participants' physical activity levels during our experiment. This method allowed us to measure the fraction of time participants remained still and performed low and medium physical activity. By low physical activity, we refer to slow movements in the space with low acceleration and moderate – movements with higher acceleration. Since participants sat at the desk for the condition with the Blockly, the hip movement is essentially zero, and we, therefore, did not add an accelerometer for comparison as it would be redundant [30].
- **Virtual Reality Sickness**: participants filled in the questions from the Simulation Sickness Questionnaire (SSQ) before and after using BlocklyVR to assess their general state of motion sickness. To calculate the SSQ score [40], we used the formula from [6]. Total SSQ scores of 20-30 reflects minimal to moderate motion sickness and greater than 40 suggest "a bad simulator" [9].

Additionally, we video-recorded user activities with BlocklyVR from outside and inside the VR headset. To further understand the impact of BlocklyVR on participants' interactions and task performance, we also measured participants' physical activity and compared it with recorded video material. Combined with the videos, the accelerometer data describe how participants moved and interacted during each task. Additionally, we conducted a thematic analysis on the video data [8] to identify distinct types of user activities. This allowed us to categorize the type of interactive user activities the participants performed second by second, e.g., interaction with the menu, and compare that data with the corresponding acceleration for that second. This allowed us to build a descriptive picture of how movement was incorporated in block-based programming activities in VR and whenever it could be beneficial. Lastly, in the end of the experiment, we collected qualitative feedback from the participants about difficulties they experienced while interacting in BlocklyVR, how VR influenced their block-based programming experience, and what they liked and disliked about BlocklyVR.

4.5 Procedure

After obtaining informed consent, we collected participants' demographic data. We then provided a brief overview of the programming environments and types of interaction. Participants familiarized themselves with a programming environment (Blockly or BlocklyVR, respectively) in a test task that required connecting two blocks so that a function "SayHi" returns "Hello World." Once the participants felt comfortable, we started experimental conditions. During the experiment, participants had to solve eight tasks, one after the other. BlocklyVR participants were asked to answer questions about the system's usability at the end of the experiment. Each experimental condition lasted approximately 30 minutes.

4.6 Data analysis

We used t-tests to compare task completion time between Blockly and BlocklyVR, given that the data was parametric. To compare differences in movements per each task, we used the one-way ANOVA as an omnibus test and t-tests for post-hoc analysis with a Bonferroni correction, given the parametric nature of the data. For the non-parametric data from the SSQ, we applied a Wilcoxon signed-rank test to compare the scores before and after interaction with BlocklyVR. For the analysis of the video recordings, one of the co-authors did a second-by-second annotation of the video feed based on the activities that participants were doing while solving the tasks. This included the following steps: (1) Looking at start second and watching a bit to see what the person is doing, (2) When a participant changed an activity, minutes and seconds from start to finish for the activity were marked on the video, (3) Annotation of the activity, e.g., changing a perspective or turning a head around,

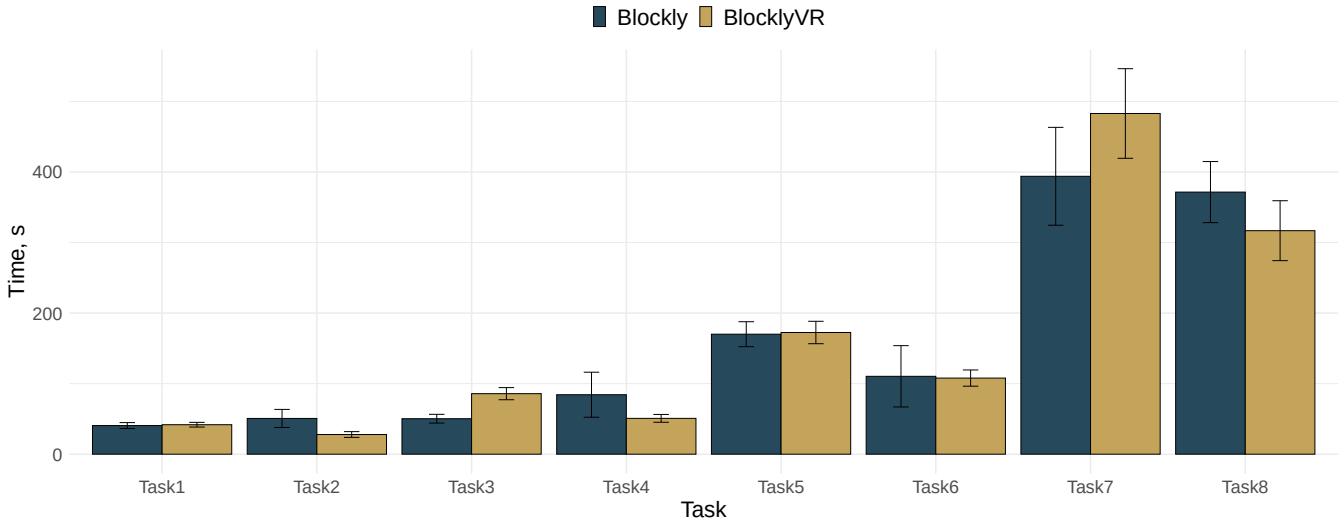

Figure 5: Task completion time per each task using Blockly and BlocklyVR.

if appropriate label did not exist yet, (4) Checking if some of annotations could be combined into the same category. These steps were repeated until finished. Lastly, for the analysis of qualitative feedback, two of the co-authors independently grouped the quotes from the participants into three groups: (1) difficulties they experienced while interacting in BlocklyVR, (2) changes in interaction while programming in Virtual Reality, and (3) aspects they liked and disliked about BlocklyVR, based on the questions we asked them after the study.

5 RESULTS

We discovered that user performance in task completion time in BlocklyVR was comparable to Blockly. Moreover, our results indicate low and moderate physical activity for participants in BlocklyVR during 28% of task time on average. Lastly, we identified four distinct types of interaction in BlocklyVR based on the video analysis. We outline these results in detail in the following subsections.

5.1 Task completion time

We discovered that participants spent a comparable amount of time-solving all tasks in Blockly or BlocklyVR ($p > 0.05$), except for Task 3. For Task 3, which required participants to sum two numbers while bringing blocks from the menu, we found that participants required more time in BlocklyVR ($M = 86sec, SD = 27$) than with Blockly ($M = 50sec, SD = 19$), as shown by a statistically significant t-test ($t(16.4) = 3.34, p < 0.01$). The summary of results is shown in Figure 5.

5.2 Physical activity duration

Our results indicate that using BlocklyVR, participants' low and medium physical activity was higher for Task 3 and for Task 5, in which participants had to match numbers and blocks, compared to the other tasks. We considered sedentary activity (or no movement) as the complement to the low and moderate physical activity, resulting in a total time consisting of no movement + low physical activity + moderate physical activity.

5.2.1 Sedentary activity. We discovered that participants remained still the least amount of time in Task 3 (task completion time was also lower in BlocklyVR for Task 3), which required finding and bringing blocks from the menu before connecting them. Task 5, which required matching numbers with functions correspondingly, also resulted in less sedentary time compared to the other tasks. The One-way ANOVA has shown statistical significance among the tasks ($F(7, 72) = 7.4, p < 0.001$). The pairwise comparisons have indicated statistically significant differences for Task 3 compared to Task 1 ($p < 0.01$), Task 2 ($p < 0.01$), Task 4 ($p < 0.01$), and Task 6 ($p < 0.01$). Additionally, we found that participants remained still more for Task 1 than Task 5 ($p < 0.05$) and Task 2 ($p < 0.01$). Lastly, participants remained still more in Task 2 than in Task 8 ($p < 0.05$).

5.2.2 Low physical activity. As for the low physical activity (i.e. slow walk), our results indicate that participants performed more low physical activity for Tasks 3 and 5, similar to the results for sedentary activity. The One-way ANOVA has shown statistical significance among the tasks ($F(7, 72) = 7.7, p < 0.001$). The pairwise comparisons have indicated statistically significant differences for Task 3 compared to Task 1 ($p < 0.001$), Task 2 ($p < 0.001$), Task 4 ($p < 0.001$), and Task 6 ($p < 0.01$). Additionally, we found that participants were performing more low physical activity for Task 5 than Task 1 ($p < 0.05$), Task 2 ($p < 0.05$), and Task 4 ($p < 0.001$). Lastly, participants' low physical activity was higher in Task 8 than in Task 2 ($p < 0.01$).

5.2.3 Moderate physical activity. In line with the above results, we found that participants' moderate physical activity was higher for Tasks 3 and 5. The One-way ANOVA has shown statistical significance among the tasks ($F(7, 72) = 4.7, p < 0.001$). The pairwise comparisons have indicated statistically significant differences for Task 3 compared to Task 1 ($p < 0.001$), Task 2 ($p < 0.001$), and Task

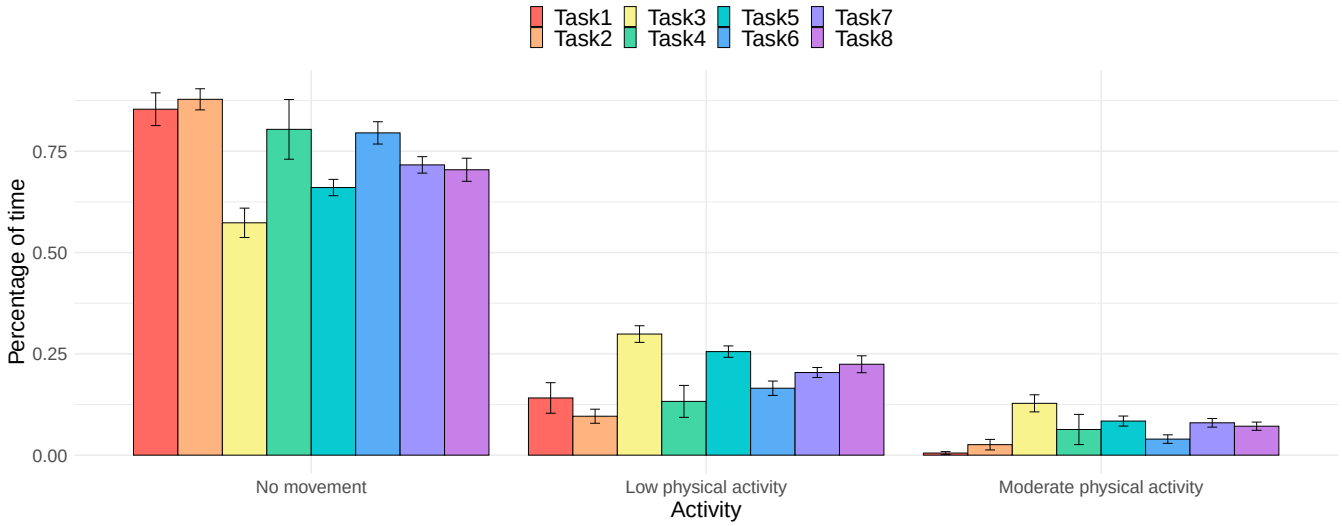

Figure 6: Movement per each task using BlocklyVR.

6 ($p < 0.05$). Lastly, participants' moderate physical activity was higher in Task 5 than in Task 1 ($p < 0.05$).

5.3 Virtual Reality Sickness

We found that virtual reality sickness was minimal after using BlocklyVR and the difference in the overall SSQ score and the sub-score of nausea, disorientation, and oculomotor was not statistically significantly different before and after using BlocklyVR ($p > 0.05$), as shown by Wilcoxon signed-rank test. The overall SSQ has increased by 2.9 points (Before: $M = 8.7, SD = 11.4$, After: $M = 11.6, SD = 11.1$), nausea decreased by 2.1 (Before: $M = 7.4, SD = 10.4$, After: $M = 5.3, SD = 5$), disorientation increased by 4.7 (Before: $M = 7.7, SD = 14.1$, After: $M = 12.4, SD = 14.7$), and oculomotor increased by 5 (Before: $M = 7.6, SD = 10$, After: $M = 12.6, SD = 12.6$).

5.4 Video observations

Based on the video observations, we identified four categories of interaction in BlocklyVR.

5.4.1 Walking from A to B. We observed participants' physical movement from points A to B related to the spatial distribution of the elements in the virtual environment. By positioning the menu and trashcans away from the task screen, participants were "forced" to walk to reach these elements. Moreover, participants had an opportunity to take multiple objects using both hands unlike interaction with a mouse in Blockly where only one object or one-handed interaction is possible. These activities accounted for 11.4% (SD:2.7%) on average of the total experiment time but 24.9% (SD:12.7%) of the average physical activity amounts (low and moderate).

5.4.2 Changing perspective. Participants spent time stepping back and forth to change their viewing perspective as a zoom-in and out function to get a better overview of the task and interaction space. Virtual reality facilitated an overview space, in which participants could easily see connections between the blocks and could look around to find necessary blocks. It accounted for 8.2% (SD: 3.5%) of the total time and 15% (SD: 4.9%) of the total physical activity.

5.4.3 2D and 3D interaction. This category accounts for 2D, e.g., pointing towards a flat surface, naming a block variable, selecting an item from a drop-down, and 3D interaction with blocks, such as moving, connecting, disconnecting, and duplicating them. This is because they account for different types of interaction within the space and can lead to different behaviors and types of movements. 2D interaction accounted for 12% (SD: 5.2%) of time and 6.15% (SD: 6.10%) of total physical activity. 3D interactions accounted for 31% (SD: 8%) of total time, and 22% (SD: 7%) of total physical activity.

5.4.4 Static activity. This category accounts for the situations in which participants did not interact with anything, e.g., they did not walk to other locations in the environment but spent time reading or thinking. It also includes movements unrelated to walking, e.g., they rotate their hips, looking around, or take small steps to readjust their body position. These activities accounted for 38.8% (SD:11.1%) of time and 20.8% (SD:7.3%) of physical activity.

5.5 Qualitative feedback

Participants stated that BlocklyVR facilitated better visibility and overview of the programming environment and liked the physical aspects. They reflected and contrasted BlocklyVR with their previous experiences using desktop interaction for textual programming. As some participant mentioned: *"I felt it was easier to make sense out of chaos. That is, if I have a lot of different ideas that I want to try, then it is easier to pick out the ones that work in VR since I can walk around them and get an overview."* [P7], *"It is pretty easy to see connections in BlocklyVR, and I think it is because of the depth of the picture and how you physically move the blocks around."* [P2]. Participants also endorsed the clear shapes and vibrant colors of the BlocklyVR blocks that improved visibility. However, it was harder

to read in BlocklyVR due to distances and occlusion from other blocks.

Participants enjoyed the ease of two-handed interaction in BlocklyVR for connecting/disconnecting blocks but commented on the slow text input. For instance, P4 noted: *"BlocklyVR was pretty easy once I grasped the controls. It was pretty efficient to tear apart components and rearrange them.". "Changing numbers and text takes a bit of time sometimes."* [P9]. Another aspect raised by the participants was the thinking process in BlocklyVR. As P10 noted: *"I believe that "thinking," i.e., coming up with the solution in your brain takes a lot of time in VR than in a normal coding environment, when I need to think about how to create an algorithm, come up with a problem solution, detect a pattern, etc. It is easier [on desktop] because I am comfortable sitting down and can switch to a pen and paper or quickly search for something in a stack overflow."* Since participants could walk and zoom in and out, they saw BlocklyVR as better-suited for progressive evaluation than their previous experience with 2D programming environments. As one participant commented: *"Literally stepping back allowed me to see the code from afar like zooming out. Using the body for that was nice"* [P10].

Four participants mentioned problems with the spatial distribution of objects in the 3D space. For example, P3 stated that: *"Losing things behind me was common, but moving to find them weren't very difficult"*. Another problem in BlocklyVR was the size of the blocks, *"Since much of the blocks are "meaningless" volume it can be difficult to interpret the code one has written because it takes up so much more physical space than regular code."* [P2]. One participant [P5] said walking to the menu to fetch new blocks felt cumbersome.

6 DISCUSSION AND FUTURE WORK

One of the questions we asked ourselves in this work: Does block-based programming become slower with the work artifacts (variables, statements, expressions) spread out in a physical, virtual, or mixed space? The findings from our study indicate that completion time was comparable between Blockly and BlocklyVR for most programming tasks. This challenges the conventional wisdom that increasing distance between objects decreases efficiency [42, 45, 56, 69]. On desktop, users can quickly move the cursor with subtle hand movements. If user interactions are more efficient on the desktop, the more interactions a user needs to do for a task, the slower the completion time will be for BlocklyVR in comparison. The video observations showed that participants spent more than 60% of the time in BlocklyVR doing user interactions (with blocks and other elements, changing views, and walking from one point in space to another). Since more than half of the time was spent interacting, if the desktop interactions were more efficient than the VR interaction, task completion times would be faster in Blockly. This suggests that the interactive techniques we used in BlocklyVR are comparable in terms of usage efficiency to the mouse and keyboard for block-based programming without introducing VR sickness. It could also suggest that some VR affordances supported BlocklyVR participants to complete the tasks efficiently. We discuss these results in detail in the following subsections.

6.1 Designing Virtual Reality Workspaces

Another question we asked was about the types of physical movements that block-based programmers would perform in such "deskless" programming environments. Our findings indicate that Virtual Reality is suitable for facilitating physical activity, e.g., active walking, as in our experiment, for a stationary task of programming with Blockly by providing an immersive 3D space. This raises a question of whether more interactivity with the whole body makes a task more engaging. By looking at the results from our experiment, the only task completed faster in Blockly than in BlocklyVR was Task 3 (Figure 5). This task also required the most physical activity, 45% on average of task time. There could therefore exist a threshold at which physical activity starts to impact task completion time. However, the task was not mentally demanding and mainly required fetching new blocks instead of problem-solving. In contrast, Task 5 required more complex problem-solving and generated physical activity at around 35% of task completion time on average. It could be that somewhere between 35-45% lies the tipping point where additional walking impedes performance. Another plausible explanation is that the intention for movement matters. In Task 3, participants only walked from the task screen to the menu, while in Task 5, participants walked to zoom out, to interact with objects, and while thinking. This could have preserved their focus better since their eyes remained on the problem [44].

The Virtual Reality sickness was minimal for solving all tasks, which makes Virtual Reality space a promising environment for future workspaces that can facilitate both solving a task but also an increased physical activity with only a VR headset and without a need to add bulky treadmill or a cycling trainer. The qualitative feedback illustrates that these physical user activities were beneficial for VR tasks. Being able to "zoom out" gave a sense of progressive evaluation, i.e., making changes and then taking a few steps back to view the "whole". This can also be achieved by zooming out with a mouse in a desktop setting. As the qualitative feedback shows, there is something special about moving your body for this action. This suggests that this ability should be considered in other VR scenarios if enough physical space is available. As Cubely [72] also suggests, using both hands in interaction was another appreciated aspect. Humans typically use their dominant hand for operations and the other hand for stability [26]. This was efficiently incorporated in BlocklyVR to disconnect blocks and something that could have contributed to comparable completion time. Lastly, both the physical aspects, i.e., moving around and interacting in a 3D space, and the visual aspect, i.e., the large screen space, were well received and, as one participant commented, made it easier to "make sense out of chaos". Future studies should explore these advantages in isolation or a different context than block-based programming. This will help us to better understand how we can potentially design future VR workspaces by possibly making them more interactive, as shown by our results.

6.2 More Movement but at What Cost?

Participants in VR had difficulty reading information on the blocks and entering text. We set the blocks' size to facilitate convenient interaction; however, smaller block sizes may function equally well. But the problem would then be the text size on the blocks, as some

comments already suggested that it was hard to read from a distance. This is a well-known problem of XR applications in general [41], but it becomes especially prevalent when incorporating a spatial work area. Reading was also a problem when blocks occluded the line of sight. Participants needed to be more accustomed to looking for information in the periphery, similar to other VR block-based interfaces [64]. Our participants did not express discomfort as in [64], likely because turning around while standing is easier than sitting down. One participant commented, however, that sitting supported thinking, and the lack of sketching with pen and paper made problem-solving harder in VR. For more complex tasks, the necessity of sketching tools is more urgent. Experimenting with virtual pens could be one way to address this issue [17, 20]. Virtual pens may also be a more efficient text-input technique than the virtual keyboard. Most negative feedback was concerned with the text-input functionality of BlocklyVR. Typing on a regular keyboard is much faster than various interaction techniques for mid-air text-input[1]. Even if VR can bring several benefits, such as walking for visibility and more physical two-handed interaction, efficient mid-air text input is crucial for bringing programming into VR, which has to be further explored in future work.

6.3 Virtual Reality for Physical Activity

Another aspect of incorporating physical activity in Virtual Reality is the potential for supporting non-sedentary interaction [32]. In other words, how can a more healthy lifestyle be integrated within typically sedentary work settings, e.g., offices. HCI researchers have conceptualized two main approaches to support physical activity: (1) in conjunction and (2) during work activities. The first approach focuses on providing tools and support in **conjunction** with office life, i.e., not at the same time as the work activity. Examples include prompts that nudge users to take active breaks [13], gamified health tracking for colleagues [13], and providing space for sports and exercise close to the office [76]. The second approach focuses on supporting physical activity **during** the work activities. Equipment such as treadmills and indoor bikes [13, 28], support tools for walking meetings [14, 27], and tangible email cards [38] are examples of such approaches. Our approach falls into the second category, integrating physical activity during a hypothetical work activity. While the use of block-based programming is scarce among professional programmers, the insights from this study could serve as a starting point for exploring physical interaction in real contexts. In the context of block-based programming, we have shown that physical activity can be used for interaction and not merely as an add-on activity independent of the task, e.g., treadmills and indoor bikes. This is especially important since movement unrelated to the users' cognitive focus can be mentally demanding [56] or distracting [44]. As for the physical activity types, most of the physical activity (75%) came from productive activities (e.g., getting an overview, 2D and 3D interactions while thinking) instead of simply walking from A to B. This explains why movement in BlocklyVR did not impact performance negatively. In future studies, exploring interactive techniques and technologies to integrate physical activity into productive activities is relevant.

We also asked a more general question on what we can learn from transferring a desk-bound activity to VR. What could the hypothetical health impact be if non-sedentary interfaces were used instead of sedentary ones? There are two forms of health hazards in our modern sedentary lifestyle. One is sitting or standing too long during the day (sedentary behavior), and the other is lack of exercise [18, 54]. In our study, participants were physically active around 28% of the time (figure 6). Most of this physical activity was classified as light, i.e., equivalent to a slow walk. In terms of reducing sedentary behavior, this light activity is sufficient for breaking up prolonged sitting or standing periods. Sedentary behavior is especially damaging after more than 7-9 hours per day [18], and minimizing daily sitting time can make a real difference. For exercise, however, moderate- or vigorous physical activity is needed. We did measure some moderate physical activity (brisk walking) in the study. Still, we believe there are probably more efficient ways of using VR to exercise, for example, through exergames [3, 76] and physical activities [33]. Intensive physical activity is likely distracting and might yield unintended side effects, such as users moving less the rest of the day because they get tired using the VR interface. When prototypes for non-sedentary interaction are mature enough to be tested in real-world scenarios, health implications should be studied in long-term case studies to control these side effects.

7 LIMITATIONS

Participants spent around 30 minutes in BlocklyVR, and a longer duration could impact fatigue. Finding when users get tired and where their focus of attention lies using eye-tracking can be tested in further studies. The study consisted of a selected set of programming tasks. This allowed us to compare completion time with Blockly. Future studies could try different tasks or try new application scenarios. For example, exploratory scenarios could be tested, where participants can program freely, e.g., programming a VR scene. We conducted our evaluation with adults with no prior experience in VR and block-based programming, and the future work should explore it for other audiences, e.g., children. Lastly, BlocklyVR requires a physical space which might only sometimes be available compared to Blockly, which can be used in a limited physical space. Thus, future work might later consider ways of comparing mouse movements in Blockly to the physical movements in BlocklyVR.

8 CONCLUSION

In this paper, we explored programming in a virtual space by transferring a system for block-based programming in Virtual Reality – BlocklyVR. To determine whether the physical movement impacts the task performance, we conducted a controlled experiment comparing it to the existing 2D version of Blockly. Our results indicate that, despite a movement-centered interface, the task completion time was comparable between Blockly and BlocklyVR, except for one task characterized by low complexity and unrelated movement. We also contribute to "desk-to-space task transfer" by identifying four distinct types of interaction in BlocklyVR. Our findings suggest implications for efficiently incorporating physical activity in productive VR interfaces, which sets a future research direction for transferring complex desk-bound intellectual tasks to Virtual Reality.

ACKNOWLEDGMENTS

We would like to thank all participants who took part in our experiment.

REFERENCES

[1] Jiban Adhikary and Keith Vertanen. 2021. Typing on Midair Virtual Keyboards: Exploring Visual Designs and Interaction Styles. In *Human-Computer Interaction – INTERACT 2021 (Lecture Notes in Computer Science)*, Carmelo Ardito, Rosa Lanzilotti, Alessio Malizia, Helen Petrie, Antonio Piccinno, Giuseppe Desolda, and Kori Inkpen (Eds.). Springer International Publishing, Cham, 132–151. https://doi.org/10.1007/978-3-030-85610-6_9

[2] Islam Almusaly, Ronald Metoyer, and Carlos Jensen. 2018. Evaluation of A Visual Programming Keyboard on Touchscreen Devices. In *2018 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*. 57–64. https://doi.org/10.1109/VLHCC.2018.8506557 ISSN: 1943-6106.

[3] Sebastian Arndt, Andrew Perkis, and Jan-Niklas Voigt-Antons. 2018. Using Virtual Reality and Head-Mounted Displays to Increase Performance in Rowing Workouts. In *Proceedings of the 1st International Workshop on Multimedia Content Analysis in Sports (MMSports'18)*. Association for Computing Machinery, New York, NY, USA, 45–50. https://doi.org/10.1145/3265845.3265848

[4] Nayeon Bak, Byeong-Mo Chang, and Kwanghoon Choi. 2020. Smart Block: A visual block language and its programming environment for IoT. *Journal of Computer Languages* 60 (Oct. 2020), 100999. https://doi.org/10.1016/j.cola.2020.100999

[5] David Bau, Jeff Gray, Caitlin Kelleher, Josh Sheldon, and Franklyn Turbak. 2017. Learnable programming: blocks and beyond. *Commun. ACM* 60, 6 (2017), 72–80. https://doi.org/10.1145/3015455

[6] Pauline Bimberg, Tim Weissker, and Alexander Kulik. 2020. On the Usage of the Simulator Sickness Questionnaire for Virtual Reality Research. In *2020 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW)*. 464–467. https://doi.org/10.1109/VRW50115.2020.00098

[7] Alan F. Blackwell. 2003. Cognitive Dimensions of Tangible Programming Languages. https://128.232.0.20/~afb21/publications/PPIG03.pdf

[8] Virginia Braun and Victoria Clarke. 2012. Thematic analysis. In *APA handbook of research methods in psychology, Vol 2: Research designs: Quantitative, qualitative, neuropsychological, and biological*. American Psychological Association, Washington, DC, US, 57–71. https://doi.org/10.1037/13620-004

[9] Polona Caserman, Augusto Garcia-Agundez, Alvar Gámez Zerban, and Stefan Göbel. 2021. Cybersickness in current-generation virtual reality head-mounted displays: systematic review and outlook. *Virtual Reality* 25, 4 (2021), 1153–1170. https://doi.org/10.1007/s10055-021-00513-6

[10] Renata Castelo-Branco, António Leitão, and Catarina Brás. 2020. Program comprehension for live algorithmic design in virtual reality. In *Conference Companion of the 4th International Conference on Art, Science, and Engineering of Programming (<Programming> '20)*. Association for Computing Machinery, New York, NY, USA, 69–76. https://doi.org/10.1145/3397537.3398475

[11] Mengyu Chen, Marko Peljhan, and Misha Sra. 2021. EntangleVR: A Visual Programming Interface for Virtual Reality Interactive Scene Generation. In *Proceedings of the 27th ACM Symposium on Virtual Reality Software and Technology (VRST '21)*. Association for Computing Machinery, New York, NY, USA, 1–6. https://doi.org/10.1145/3489849.3489872

[12] Will Crichton, Maneesh Agrawala, and Pat Hanrahan. 2021. The Role of Working Memory in Program Tracing. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems (CHI '21)*. Association for Computing Machinery, New York, NY, USA, 1–13. https://doi.org/10.1145/3411764.3445257

[13] Ida Damen, Hans Brombacher, Carine Lallemand, Rens Brankaert, Aarnout Brombacher, Pieter van Wesemael, and Steven Vos. 2020. A Scoping Review of Digital Tools to Reduce Sedentary Behavior or Increase Physical Activity in Knowledge Workers. *International Journal of Environmental Research and Public Health* 17, 2 (Jan. 2020), 499. https://doi.org/10.3390/ijerph17020499 Number: 2 Publisher: Multidisciplinary Digital Publishing Institute.

[14] Ida Damen, Carine Lallemand, Rens Brankaert, Aarnout Brombacher, Pieter van Wesemael, and Steven Vos. 2020. Understanding Walking Meetings: Drivers and Barriers. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. ACM, Honolulu HI USA, 1–14. https://doi.org/10.1145/3313831.3376141

[15] Kevin Doherty and Gavin Doherty. 2018. Engagement in HCI: Conception, Theory and Measurement. *Comput. Surveys* 51, 5 (Nov. 2018), 99:1–99:39. https://doi.org/10.1145/3234149

[16] James Dominic, Brock Tubre, Jada Houser, Charles Ritter, Deborah Kunkel, and Paige Rodeghero. 2020. Program Comprehension in Virtual Reality. In *Proceedings of the 28th International Conference on Program Comprehension (ICPC '20)*. Association for Computing Machinery, New York, NY, USA, 391–395. https://doi.org/10.1145/3387904.3389287

[17] Tobias Drey, Jan Gugenheimer, Julian Karlbauer, Maximilian Milo, and Enrico Rukzio. 2020. VRSketchIn: Exploring the Design Space of Pen and Tablet Interaction for 3D Sketching in Virtual Reality. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems (CHI '20)*. Association for Computing Machinery, New York, NY, USA, 1–14. https://doi.org/10.1145/3313831.3376628

[18] Elin Ekblom Bak. 2021. *Långvarigt stillasittande : en hälsofara i tiden?* Studentlitteratur AB. http://urn.kb.se/resolve?urn=urn:nbn:se:gih:diva-6631

[19] Anthony Elliott, Brian Peiris, and Chris Parnin. 2015. Virtual Reality in Software Engineering: Affordances, Applications, and Challenges. In *2015 IEEE/ACM 37th IEEE International Conference on Software Engineering*. IEEE, Florence, Italy, 547–550. https://doi.org/10.1109/ICSE.2015.191

[20] Hesham Elsayed, Mayra Donaji Barrera Machuca, Christian Schaarschmidt, Karola Marky, Florian Müller, Jan Riemann, Andrii Matviienko, Martin Schmitz, Martin Weigel, and Max Mühlhäuser. 2020. VRSketchPen: Unconstrained Haptic Assistance for Sketching in Virtual 3D Environments. In *Proceedings of the 26th ACM Symposium on Virtual Reality Software and Technology* (Virtual Event, Canada) *(VRST '20)*. Association for Computing Machinery, New York, NY, USA, Article 3, 11 pages. https://doi.org/10.1145/3385956.3418953

[21] João Marcelo Evangelista Belo, Anna Maria Feit, Tiare Feuchtner, and Kaj Grønbæk. 2021. XRgonomics: Facilitating the Creation of Ergonomic 3D Interfaces. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems (CHI '21)*. Association for Computing Machinery, New York, NY, USA, 1–11. https://doi.org/10.1145/3411764.3445349

[22] Ylva Fernaeus and Jakob Tholander. 2006. Finding design qualities in a tangible programming space. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '06)*. Association for Computing Machinery, New York, NY, USA, 447–456. https://doi.org/10.1145/1124772.1124839

[23] Michael H. Fischer. 2016. Inception: a creative coding environment for virtual reality, in virtual reality. In *Proceedings of the 22nd ACM Conference on Virtual Reality Software and Technology (VRST '16)*. Association for Computing Machinery, New York, NY, USA, 339–340. https://doi.org/10.1145/2993369.2996354

[24] Joshua Fu, Ryan Lim, Nasser Giacaman, and Craig J. Sutherland. 2021. KareNao: A Tangible Block-Based Programming Environment. In *2021 18th International Conference on Ubiquitous Robots (UR)*. 314–319. https://doi.org/10.1109/UR52253.2021.9494672 ISSN: 2325-033X.

[25] Bryson Goolsby, Dianne Pawluk, Hyun Woo Kim, and Giovanni Fusco. 2021. A Tangible Block Editor for the Scratch Programming Language. In *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems (CHI EA '21)*. Association for Computing Machinery, New York, NY, USA, 1–6. https://doi.org/10.1145/3411763.3451833

[26] Yves Guiard. 1987. Asymmetric Division of Labor in Human Skilled Bimanual Action. *Journal of Motor Behavior* 19, 4 (Dec. 1987), 486–517. https://doi.org/10.1080/00222895.1987.10735426 Publisher: Routledge _eprint: https://doi.org/10.1080/00222895.1987.10735426.

[27] Luke Haliburton, Natalia Bartłomiejczyk, Albrecht Schmidt, Paweł W. Woźniak, and Jasmin Niess. 2023. The Walking Talking Stick: Understanding Automated Note-Taking in Walking Meetings. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems (CHI '23)*. Association for Computing Machinery, New York, NY, USA, 1–16. https://doi.org/10.1145/3544548.3580986

[28] Luke Haliburton and Albrecht Schmidt. 2020. Technologies for healthy work. *Interactions* 27, 3 (April 2020), 64–66. https://doi.org/10.1145/3386391

[29] Brian Harvey, Daniel D. Garcia, Tiffany Barnes, Nathaniel Titterton, Daniel Armendariz, Luke Segars, Eugene Lemon, Sean Morris, and Josh Paley. 2013. SNAP! (build your own blocks) (abstract only). In *Proceeding of the 44th ACM technical symposium on Computer science education (SIGCSE '13)*. Association for Computing Machinery, New York, NY, USA, 759. https://doi.org/10.1145/2445196.2445507

[30] William L. Haskell, I-min Lee, Russel R. Pate, and Kenneth E Powell. 2007. Physical Activity and Public Health. *Circulation* 116, 9 (Aug. 2007), 1081–1093. https://doi.org/10.1161/CIRCULATIONAHA.107.185649 Publisher: American Heart Association.

[31] Wilhelm Hasselbring, Alexander Krause, and Christian Zirkelbach. 2020. ExplorViz: Research on software visualization, comprehension and collaboration. *Software Impacts* 6 (Nov. 2020), 100034. https://doi.org/10.1016/j.simpa.2020.100034

[32] Martin Hedlund, Cristian Bogdan, and Gerrit Meixner. 2022. Creating a Post-sedentary Work Context for Software Engineering. In *Sense, Feel, Design (Lecture Notes in Computer Science)*, Carmelo Ardito, Rosa Lanzilotti, Alessio Malizia, Marta Larusdottir, Lucio Davide Spano, José Campos, Morten Hertzum, Tilo Mentler, José Abdelnour Nocera, Lara Piccolo, Stefan Sauer, and Gerrit van der Veer (Eds.). Springer International Publishing, Cham, 123–138. https://doi.org/10.1007/978-3-030-98388-8_12

[33] Martin Hedlund, Anders Lundström, Cristian Bogdan, and Andrii Matviienko. 2023. Jogging-in-Place: Exploring Body-Steering Methods for Jogging in Virtual Environments. In *Proceedings of the 22nd International Conference on Mobile and Ubiquitous Multimedia* (Vienna, Austria) *(MUM '23)*. Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/3626705.3627778

[34] Hiroshi Ishii. 2008. Tangible bits: beyond pixels. In *Proceedings of the 2nd international conference on Tangible and embedded interaction (TEI '08)*. Association for Computing Machinery, New York, NY, USA, xv–xxv. https://doi.org/10.1145/1347390.1347392

[35] Chris Johnson and Peter Bui. 2015. Blocks in, blocks out: A language for 3D models. In *2015 IEEE Blocks and Beyond Workshop (Blocks and Beyond)*. 77–82. https://doi.org/10.1109/BLOCKS.2015.7369007

[36] Joseph J. LaViola Jr, Ernst Kruijff, Ryan P. McMahan, Doug Bowman, and Ivan P. Poupyrev. 2017. *3D User Interfaces: Theory and Practice*. Addison-Wesley Professional. Google-Books-ID: fxWSDgAAQBAJ.

[37] Dominic Kao, Christos Mousas, Alejandra J. Magana, D. Fox Harrell, Rabindra Ratan, Edward F. Melcer, Brett Sherrick, Paul Parsons, and Dmitri A. Gusev. 2020. Hack.VR: A Programming Game in Virtual Reality. *arXiv:2007.04495 [cs]* (Nov. 2020). http://arxiv.org/abs/2007.04495 arXiv: 2007.04495.

[38] Philip Keller, Roy van den Heuvel, and Carine Lallemand. 2023. Bringing Movement to Digital Tasks at the Office: Designing an Acceptably Active Interface Interaction for Sending Emails. In *Proceedings of the Seventeenth International Conference on Tangible, Embedded, and Embodied Interaction (TEI '23)*. Association for Computing Machinery, New York, NY, USA, 1–8. https://doi.org/10.1145/3569009.3573113

[39] Annie Kelly, R. Benjamin Shapiro, Jonathan de Halleux, and Thomas Ball. 2018. ARcadia: A Rapid Prototyping Platform for Real-time Tangible Interfaces. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. Association for Computing Machinery, New York, NY, USA, 1–8. https://doi.org/10.1145/3173574.3173983

[40] Robert S. Kennedy, Norman E. Lane, Kevin S. Berbaum, and Michael G. Lilienthal. 1993. Simulator Sickness Questionnaire: An Enhanced Method for Quantifying Simulator Sickness. *The International Journal of Aviation Psychology* 3, 3 (1993), 203–220. https://doi.org/10.1207/s15327108ijap0303_3

[41] Veronika Krauß, Michael Nebeling, Florian Jasche, and Alexander Boden. 2022. Elements of XR Prototyping: Characterizing the Role and Use of Prototypes in Augmented and Virtual Reality Design. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems (CHI '22)*. Association for Computing Machinery, New York, NY, USA, 1–18. https://doi.org/10.1145/3491102.3517714

[42] William Lidwell, Kritina Holden, and Jill Butler. 2010. *Universal Principles of Design, Revised and Updated: 125 Ways to Enhance Usability, Influence Perception, Increase Appeal, Make Better Design Decisions, and Teach Through Design*. Rockport Publishers. Google-Books-ID: 3RFyaF7jCZsC.

[43] Yuhan Lin and David Weintrop. 2021. The landscape of Block-based programming: Characteristics of block-based environments and how they support the transition to text-based programming. *Journal of Computer Languages* 67 (Dec. 2021), 101075. https://doi.org/10.1016/j.cola.2021.101075

[44] Simon P. Liversedge and John M. Findlay. 2000. Saccadic eye movements and cognition. *Trends in Cognitive Sciences* 4, 1 (Jan. 2000), 6–14. https://doi.org/10.1016/S1364-6613(99)01418-7

[45] I. Scott MacKenzie. 1992. Fitts' Law as a Research and Design Tool in Human-Computer Interaction. *Human–Computer Interaction* 7, 1 (March 1992), 91–139. https://doi.org/10.1207/s15327051hci0701_3 Publisher: Taylor & Francis _eprint: https://doi.org/10.1207/s15327051hci0701_3.

[46] Alessio Malizia, Tommaso Turchi, and Kai A. Olsen. 2017. Block-oriented programming with tangibles: An engaging way to learn computational thinking skills. In *2017 IEEE Blocks and Beyond Workshop (B&B)*. 61–64. https://doi.org/10.1109/BLOCKS.2017.8120413

[47] J. Maloney, L. Burd, Y. Kafai, N. Rusk, B. Silverman, and M. Resnick. 2004. Scratch: a sneak preview [education]. In *Proceedings. Second International Conference on Creating, Connecting and Collaborating through Computing, 2004.* 104–109. https://doi.org/10.1109/C5.2004.1314376

[48] Paul Marshall. 2007. Do tangible interfaces enhance learning?. In *Proceedings of the 1st international conference on Tangible and embedded interaction (TEI '07)*. Association for Computing Machinery, New York, NY, USA, 163–170. https://doi.org/10.1145/1226969.1227004

[49] Andy Matuschak, Michael Nielsen, Andy Matuschak, and Michael Nielsen. 2019. How can we develop transformative tools for thought? (2019). https://numinous.productions/ttft

[50] Andrii Matviienko, Marcel Langer, Florian Müller, Martin Schmitz, and Max Mühlhäuser. 2021. VRtangibles: Assisting Children in Creating Virtual Scenes Using Tangible Objects and Touch Input. In *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems* (Yokohama, Japan) *(CHI EA '21)*. Association for Computing Machinery, New York, NY, USA, Article 460, 7 pages. https://doi.org/10.1145/3411763.3451671

[51] Timothy S. McNerney. 1999. *Tangible programming bricks : an approach to making programming accessible to everyone*. Thesis. Massachusetts Institute of Technology. https://dspace.mit.edu/handle/1721.1/62094 Accepted: 2011-04-04T17:39:09Z.

[52] Edward Melcer. 2017. Moving to Learn: Exploring the Impact of Physical Embodiment in Educational Programming Games. In *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems (CHI EA '17)*. Association for Computing Machinery, New York, NY, USA, 301–306. https://doi.org/10.1145/3027063.3027129

[53] Luke Moors, Andrew Luxton-Reilly, and Paul Denny. 2018. Transitioning from Block-Based to Text-Based Programming Languages. In *2018 International Conference on Learning and Teaching in Computing and Engineering (LaTICE)*. 57–64. https://doi.org/10.1109/LaTICE.2018.000-5 ISSN: 2475-1057.

[54] World Health Organization. 2019. *Global Action Plan on Physical Activity 2018-2030: More Active People for a Healthier World*. World Health Organization. Google-Books-ID: RnOyDwAAQBAJ.

[55] Mitchel Resnick and Brian Silverman. 2005. Some reflections on designing construction kits for kids. In *Proceedings of the 2005 conference on Interaction design and children (IDC '05)*. Association for Computing Machinery, New York, NY, USA, 117–122. https://doi.org/10.1145/1109540.1109556

[56] David P. Richardson, John J. Foxe, Kevin A. Mazurek, Nicholas Abraham, and Edward G. Freedman. 2022. Neural markers of proactive and reactive cognitive control are altered during walking: A Mobile Brain-Body Imaging (MoBI) study. *NeuroImage* 247 (Feb. 2022), 118853. https://doi.org/10.1016/j.neuroimage.2021.118853

[57] A. Santos-Lozano, F. Santín-Medeiros, G. Cardon, G. Torres-Luque, R. Bailón, C. Bergmeir, J. R. Ruiz, A. Lucia, and N. Garatachea. 2013. Actigraph GT3X: Validation and Determination of Physical Activity Intensity Cut Points. *International Journal of Sports Medicine* 34, 11 (Nov. 2013), 975–982. https://doi.org/10.1055/s-0033-1337945 Publisher: © Georg Thieme Verlag KG.

[58] Ronny Scherer, Fazilat Siddiq, and Bárbara Sánchez Viveros. 2020. A meta-analysis of teaching and learning computer programming: Effective instructional approaches and conditions. *Computers in Human Behavior* 109 (Aug. 2020), 106349. https://doi.org/10.1016/j.chb.2020.106349

[59] Albrecht Schmidt. 2016. Increasing Computer Literacy with the BBC micro:bit. *IEEE Pervasive Computing* 15, 2 (April 2016), 5–7. https://doi.org/10.1109/MPRV.2016.23 Conference Name: IEEE Pervasive Computing.

[60] Rafael J. Segura, Francisco J. del Pino, Carlos J. Ogáyar, and Antonio J. Rueda. 2020. VR-OCKS: A virtual reality game for learning the basic concepts of programming. *Computer Applications in Engineering Education* 28, 1 (2020), 31–41. https://doi.org/10.1002/cae.22172 _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/cae.22172.

[61] Hong SeongYong and Hwang YongHyun. 2020. DESIGN AND IMPLEMENTATION FOR IORT BASED REMOTE CONTROL ROBOT USING BLOCK-BASED PROGRAMMING. *Issues In Information Systems* (2020). https://doi.org/10.48009/4_iis_2020_317-330

[62] BEN SHNEIDERMAN. 1982. The future of interactive systems and the emergence of direct manipulation. *Behaviour & Information Technology* 1, 3 (July 1982), 237–256. https://doi.org/10.1080/01449298208914450 Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/01449298208914450.

[63] Ben Shneiderman, Gerhard Fischer, Mary Czerwinski, Mitch Resnick, Brad Myers, Linda Candy, Ernest Edmonds, Mike Eisenberg, Elisa Giaccardi, Tom Hewett, Pamela Jennings, Bill Kules, Kumiyo Nakakoji, Jay Nunamaker, Randy Pausch, Ted Selker, Elisabeth Sylvan, and Michael Terry. 2006. Creativity Support Tools: Report From a U.S. National Science Foundation Sponsored Workshop. *International Journal of Human–Computer Interaction* 20, 2 (May 2006), 61–77. https://doi.org/10.1207/s15327590ijhc2002_1 Publisher: Taylor & Francis _eprint: https://doi.org/10.1207/s15327590ijhc2002_1.

[64] Robert Sims, Nathan Rutherford, Prashanthy Sukumaran, Nikola Yotov, Thomas Smith, and Abhijit Karnik. 2021. Logibot: Investigating Engagement and Development of Computational Thinking Through Virtual Reality. In *2021 7th International Conference of the Immersive Learning Research Network (iLRN)*. 1–5. https://doi.org/10.23919/iLRN52045.2021.9459352

[65] Isabelle M. L. Souza, Wilkerson L. Andrade, Lívia M. R. Sampaio, and Ana Liz Souto O. Araujo. 2018. A Systematic Review on the use of LEGO® Robotics in Education. In *2018 IEEE Frontiers in Education Conference (FIE)*. 1–9. https://doi.org/10.1109/FIE.2018.8658751 ISSN: 2377-634X.

[66] Marco Speicher, Anna Maria Feit, Pascal Ziegler, and Antonio Krüger. 2018. Selection-based Text Entry in Virtual Reality. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. ACM, Montreal QC Canada, 1–13. https://doi.org/10.1145/3173574.3174221

[67] Nanlin Sun, Annette Feng, Ryan Patton, Yotam Gingold, and Wallace Lages. 2021. Programmable Virtual Reality Environments. In *2021 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW)*. 619–620. https://doi.org/10.1109/VRW52623.2021.00192

[68] Kazuki Tada and Jiro Tanaka. 2015. Tangible Programming Environment Using Paper Cards as Command Objects. *Procedia Manufacturing* 3 (Jan. 2015), 5482–5489. https://doi.org/10.1016/j.promfg.2015.07.693

[69] Eleftherios Triantafyllidis and Zhibin Li. 2021. The Challenges in Modeling Human Performance in 3D Space with Fitts' Law. In *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems (CHI EA '21)*. Association for Computing Machinery, New York, NY, USA, 1–9. https://doi.org/10.1145/3411763.3443442

[70] Tommaso Turchi and Alessio Malizia. 2016. Fostering computational thinking skills with a tangible blocks programming environment. In *2016 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*. 232–233. https://doi.org/10.1109/VLHCC.2016.7739692 ISSN: 1943-6106.

[71] Robert Twomey, Tommy Sharkey, Timothy Wood, Amy Eguchi, Monica Sweet, and Ying Choon Wu. 2022. An Immersive Environment for Embodied Code. In *Extended Abstracts of the 2022 CHI Conference on Human Factors in Computing Systems (CHI EA '22)*. Association for Computing Machinery, New York, NY, USA, 1–4. https://doi.org/10.1145/3491101.3519896

[72] Juraj Vincur, Martin Konopka, Jozef Tvarozek, Martin Hoang, and Pavol Navrat. 2017. Cubely: virtual reality block-based programming environment. In *Proceedings of the 23rd ACM Symposium on Virtual Reality Software and Technology (VRST '17)*. Association for Computing Machinery, New York, NY, USA, 1–2. https://doi.org/10.1145/3139131.3141785

[73] Kaisa Väänänen-Vainio-Mattila, Katja Suhonen, Jari Laaksonen, Johan Kildal, and Koray Tahiroğlu. 2013. User experience and usage scenarios of audio-tactile interaction with virtual objects in a physical environment. In *Proceedings of the 6th International Conference on Designing Pleasurable Products and Interfaces (DPPI '13)*. Association for Computing Machinery, New York, NY, USA, 67–76. https://doi.org/10.1145/2513506.2513514

[74] Torben Wallbaum, Swamy Ananthanarayan, Andrii Matviienko, and Susanne Boll. 2020. A Real-Time Distributed Toolkit to Ease Children's Exploration of IoT. In *Proceedings of the 11th Nordic Conference on Human-Computer Interaction: Shaping Experiences, Shaping Society* (Tallinn, Estonia) *(NordiCHI '20)*. Association for Computing Machinery, New York, NY, USA, Article 9, 9 pages. https://doi.org/10.1145/3419249.3420179

[75] David Weintrop, David C. Shepherd, Patrick Francis, and Diana Franklin. 2017. Blockly goes to work: Block-based programming for industrial robots. In *2017 IEEE Blocks and Beyond Workshop (B&B)*. 29–36. https://doi.org/10.1109/BLOCKS.2017.8120406

[76] Soojeong Yoo, Phillip Gough, and Judy Kay. 2020. Embedding a VR Game Studio in a Sedentary Workplace: Use, Experience and Exercise Benefits. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems (CHI '20)*. Association for Computing Machinery, New York, NY, USA, 1–14. https://doi.org/10.1145/3313831.3376371

[77] Lei Zhang and Steve Oney. 2020. FlowMatic: An Immersive Authoring Tool for Creating Interactive Scenes in Virtual Reality. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology*. ACM, Virtual Event USA, 342–353. https://doi.org/10.1145/3379337.3415824