A General Reactive Algorithm for Redirected Walking using Artificial Potential Functions

Jerald Thomas* Evan Suma Rosenberg†

University of Minnesota

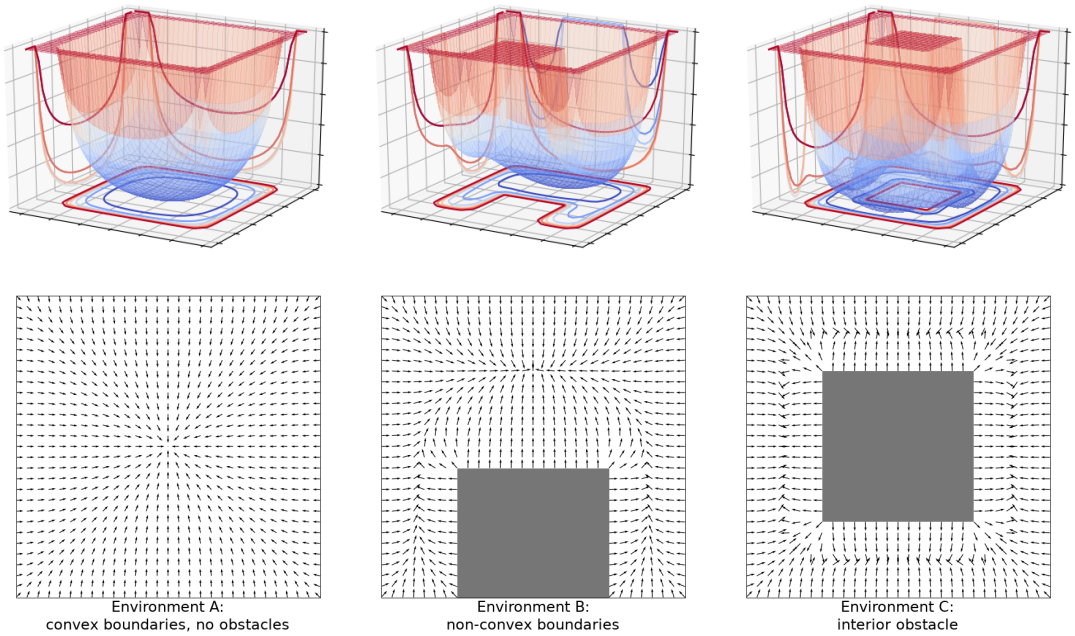

Figure 1: Artificial potential function surfaces (top) and normalized negative gradient fields (bottom) of the three evaluated physical environments. The proposed reactive redirected walking algorithm utilizes the negative gradient as a direction in which to steer the user.

Abstract

Redirected walking enables users to locomote naturally within a virtual environment that is larger than the available physical space. These systems depend on steering algorithms that continuously redirect users within limited real world boundaries. While a majority of the most recent research has focused on predictive algorithms, it is often necessary to utilize reactive approaches when the user's path is unconstrained. Unfortunately, previously proposed reactive algorithms assume a completely empty space with convex boundaries and perform poorly in complex real world spaces containing obstacles. To overcome this limitation, we present Push/Pull Reactive (P2R), a novel algorithm that uses an artificial potential function to steer users away from potential collisions. We also introduce three new reset strategies and conducted an experiment to evaluate which one performs best when used with P2R. Simulation results demonstrate that the proposed approach outperforms the previous state-of-the-art reactive algorithm in non-convex spaces with and without interior obstacles.

Index Terms: Human-centered computing—Human computer interaction (HCI)—Interaction paradigms—Virtual reality;

*e-mail: thoma891@umn.edu

†e-mail: suma@umn.edu

1 Introduction

Redirected walking (RDW) enables natural movement in immersive virtual environments that are smaller than the available physical space. Compared to less immersive locomotion techniques such as "flying," walking has been shown to improve users' abilities to navigate [15] and their subjective sense of presence [21]. Algorithms that employ RDW are generally categorized as either reactive or predictive. Reactive algorithms have no knowledge of the user's intended virtual path and instead rely on simple heuristics. For example, Steer-to-Center (S2C) attempts to steer the user towards the center of the physical space at all times. Predictive algorithms have some knowledge of the user's intended virtual path. With this knowledge, predictive algorithms are able to steer the user in a way that is favorable considering the user's probable future motion. In cases where predictive algorithms are applicable, they have been shown to greatly out-perform reactive algorithms [10, 22]. However, due to their relatively simplistic implementation, reactive algorithms are more generalizable for applications that involve free exploration or unconstrained movement. Therefore, there is no "silver bullet" approach to redirected walking, and improvements in both reactive and predictive algorithms are valuable for advancing the state-of-the-art in virtual reality locomotion.

Current reactive RDW algorithms rely on three assumptions about the physical environment: that it is clear of any obstacles, convex

(usually rectangular or circular), and static. In this paper, we refer to spaces that meet these assumptions as *ideal* physical environments, and those that do not as *non-ideal* physical environments. In most potential operating environments outside of a tightly controlled lab or exhibit, these assumptions are overly restrictive and unrealistic for practical use "in the wild." To solve this, RDW algorithms should be contextually aware of the physical environment and able to generate a solution that avoids obstacles while maximizing use of physical spaces with non-convex boundaries.

To address these goals, this work presents a novel reactive approach that was designed to relax the assumptions of ideal physical environments, known as the Push/Pull Reactive (P2R) algorithm. P2R utilizes potential functions that include an attractive force component and a repulsive force component. Essentially, it works by pushing the user away from undesirable locations (boundaries and obstacles) and pulling the user towards desirable locations (goals).

This paper presents four primary contributions. The first is a formal description of the P2R algorithm. Second, we describe three new reset techniques that are also necessary for applying RDW in non-ideal spaces. We then conducted a simulation-based experiment to determine which of these reset strategies will perform best when utilized with P2R. Finally, we present an evaluation that showing that P2R outperforms the previous state-of-the-art reactive algorithm in non-ideal physical environments.

2 Background and Related Works

Over the past 15 years, there has been an extensive quantity literature on redirected walking in virtual reality; a recent review can be found in Nilsson et al. [13]. RDW employs three main techniques that are commonly referred to as self-motion gains [14]. These gains exploit the fact that human vision dominates vestibular sensation when the magnitude of the conflict does not exceed perceptual thresholds, thereby enabling users to be imperceptibly redirected into walking along virtual path that is decoupled from their physical path. Translation gains apply a scale factor to the user's positional displacement. Rotation gain is similar to translation gain, but it is applied to the user's orientation instead of positional displacement. It is assumed that the system cannot always find a set of gains that maintain perceptual thresholds while keeping the user within the boundaries of the physical environment. Therefore, when the user approaches the boundaries, the system performs a reorientation event. Typically referred to resets, these events essentially pause the virtual experience and reorients the user in a favorable directly, usually towards the center in an ideal physical environment.

Perceptual Thresholds

Redirected walking can provide a compelling experience if the user does not perceive the manipulations induced by the system. Steinicke et al. determined that in order to remain undetected by the average user, rotation gains need to be greater than 0.67 and less than 1.24, translation gains need to be greater than 0.86 and less than 1.26, and curvature gains need to have at least a 22.03 meter radius [19]. Grechkin et al. showed that combining translation and curvature gain does not appear to change the user's detection threshold for curvature gain [4]. Neth et al. demonstrated a correlation between the user's physical velocity and curvature gain detection, namely that as the user physically translates faster, their ability to detect curvature gain is greater [11]. Hutton et al. noted that the methods previously used to determine perceptual thresholds suggested large amounts of variability between participants. They introduced a method for estimating an individual's perceptual thresholds for rotation gains that is much quicker than previous methods [7]. Recently, there has been work attempting to apply rotation and curvature gains that exceed the usual perceptual limits by introducing rotations during blinking or saccadic eye movements [9, 12, 20].

Reactive Algorithms

Some of the first RDW algorithms were reactive. Hodgson et al. showed that in most scenarios, S2C outperforms other reactive algorithms [6]. They posit that steer-to-orbit (S2O), a reactive RDW algorithm that tries to steer the user along a circle around the physical environment center, might outperform S2C if the virtual path is long and consists of very few turns. Azmandian et al. further compared reactive algorithms in a variety physical environment sizes and aspect ratios [1]. They also modified the S2C algorithm by adding "center-based translation gain." Their results reinforce those found by Hodgson et al., showing that S2C (with center-based translation gain) outperforms the other reactive algorithms in most practical use cases. However, for a 1000m straight virtual path, S2O outperformed S2C in a narrow window of physical environment sizes that corresponded to the maximum curvature gain. Initial descriptions of RDW algorithms for non-ideal physical environments have recently been proposed by Chen et al., though no evaluation was performed [3].

Obstacles in the Physical Environment

An essential step for environmentally aware RDW algorithms is acquiring obstacle and boundary information. The simplest method is to provide this information before the experience begins, but this is potentially very tedious and would not allow for dynamic obstacles. Hirt et al. provided a method for extracting environment geometry in real-time [5]. Their method uses simultaneous localization and mapping (SLAM) techniques to obtain environment geometry as the user explores the physical environment. This method can acquire environment information either before or during the experience and allows for dynamic obstacles to be registered. SLAM tracking capabilities are becoming increasingly integrated with immersive displays and mobile devices, which is a promising trend for the viability of RDW algorithms that require knowledge of the physical environment.

Other methods for managing environments with obstacles in virtual reality that do not employ RDW have been evaluated. Simeone et al. represented physical obstacles and boundaries with virtual objects that were contextually relevant to the virtual experience [17]. The study found that users are more likely to avoid walking into obstacles but were also likely to attempt to interact with the virtual object. The same group also evaluated the effectiveness of replacing obstacles and boundaries with surfaces that a person would not typically walk onto, such as water or lava, and found that they could successfully alter a user's path to avoid obstacles [16]. Sra et al. presented similar work in which they created procedurally generated virtual environments that had a walkable area that matched the walkable area of the physical environment. Their system was capable of recognizing and tracking a small set of physical objects, for example a chair, and replacing them in the virtual environments with proper representations. This allowed the user to interact with certain physical objects [18].

3 P2R Algorithm Description

In terms of implementation details, P2R is similar to S2C with center-based translation gain. The main difference is in determining the ideal direction to steer the user. S2C relies on the simple center-based heuristic, which is chosen as a goal position because it is the farthest position from all boundaries in an ideal rectangular physical space. However, for more complex physical environments containing obstacles or irregular boundary shapes, the center may no longer be the most ideal position to steer the user. It is impractical to pre-compute an ideal goal position for all possible environment layouts. Additionally, dynamic environments with moving obstacles will alter the ideal goal position. As such, an algorithm needs to be able to determine an ideal position in real-time. Comparable problems exist in the field of robotics and are commonly solved

using artificial potential functions [8]. Equation 1 shows a simple example of a potential function given the set of obstacles O. Here. the value of the potential function at a user's position x is one half the distance to the goal position plus the sum of one over the distance to the nearest point of an obstacle for all obstacles.

$$U(x) = \frac{1}{2}||x - x_{goal}|| + \sum_{ob \in O} \frac{1}{||x - x_{ob}||} \quad (1)$$

Equation 1 can be broken into two components: an attractive force (Equation 2) and a repulsive force (Equation 3).

$$U_{attractive}(x) = \frac{1}{2}||x - x_{goal}|| \quad (2)$$

$$U_{repulsive}(x) = \sum_{ob \in O} \frac{1}{||x - x_{ob}||} \quad (3)$$

Instead of using a goal heuristic to keep the user away from obstacles and the boundary, P2R uses the repulsive force component of a potential function. This allows the possibility to have a goal position that does not necessarily have the purpose of keeping the user away from the boundaries. For example, one may use the attractive force component to steer the user towards a physical object they wish the user to interact with. This work focused on comparing P2R with S2C, so the attractive force component was not used.

Setup for P2R consists of providing the algorithm with RDW gain thresholds and a list of obstacles. For the implementation, the evaluations the obstacles had a convexity requirement. The boundaries of each environment were modeled as four rectangular obstacles.

Each frame, P2R uses the centered finite difference method to calculate the negative gradient, $-\nabla U(x)$, of the potential function at the user's position, x. The algorithm then uses the negative gradient to assign RDW gains for the frame. When the user is translating, the algorithm calculates curvature gain and translation gain. If the negative gradient is clockwise of the user's forward direction, the algorithm applies maximum positive curvature. If it is counter-clockwise, maximum negative curvature is applied. If the dot product of the negative gradient and the user's forward direction is negative, the algorithm applies minimum translation gain. This effectively slows down the user when the user is translating in a direction opposite of the negative gradient. When the user is rotating, the algorithm calculates rotation gain. If the user is rotating in such a way that the angle between the user's forward direction and the negative gradient is decreasing, then maximum rotation gain is applied. If the angle is increasing, then minimum rotation gain is applied.

A run-time complexity analysis shows that the P2R algorithm is $O(N)$ where N is the number of obstacles in the environment.

4 Reset Strategies

Obstacles present a new problem to the standard "reset to center" (R2C) reset strategy that has previously been applied in ideal physical spaces. For example, if the obstacle that has caused the user to reset is positioned between the user and the center of the physical environment, this strategy will fail. In this scenario, the user would complete the reset still facing the obstacle, which would either result in a collision or an endless reset loop. To overcome this limitation, we propose three possible alternative reset strategies: modified reset to center (MR2C), reset to gradient (R2G), and step-forward reset to gradient (SFR2G).

MR2C: When the user's reset is triggered by an environment boundary or by an obstacle that is not positioned between the user and the physical environment center, MR2C behaves exactly as R2C. However, when the reset is triggered by an obstacle positioned between the user and the physical environment center, the user is reoriented parallel to the face of the obstacle in the closet direction towards the center.

R2G: Given a potential function and its negative gradient, the basic assumption is that the negative gradient points towards the most ideal direction. Therefore, this method reorients the user in the direction of the negative gradient at the current position.

SFR2G: There are situations where reorienting the user in the direction of the negative gradient may direct them towards another obstacle or boundary instead of open space. This is particularly true when there is no attractive force component of the potential function. To accommodate for this, SFR2G takes several small steps following gradient descent along the potential function and uses the resulting position as a target to reorient the user. Step size and number of steps are variables that can be tuned.

5 Evaluation Framework

5.1 Physical Environment Layouts

For the following experiments there were three physical environment layouts: environments A, B, and C. All layouts have square boundaries with a side length of either 10m or 20m. Environment A is a control layout and consists of an environment free of any obstacles. Environments B and C each contain a single square obstacle with a side length of 40% of the environment side length. The obstacle in environment B was adjacent to the lower y boundary and centered along the x boundary, essentially creating a non-convex physical environment. The obstacle in environment C was centered to the physical environment in both dimensions. These environments were chosen as a representative sample of non-ideal physical environments. Environment B represents a space with non-convex boundaries, and environment C represents a space with an interior obstacle. Because the obstacle for environment C was chosen to be in the center, it represents the worst case scenario for S2C.

5.2 Simulation Design

The two experiments reported in this paper were conducted using simulation. To comprehensively evaluate the performance of RDW algorithms, it is often necessary to run a very large number of trials and test numerous possible parameters, which is impractical for live user studies. For this reason, simulation-based evaluation is a common practice in the RDW literature (e.g., [1, 2, 6, 22]). Each permutation consisted of 100 trials, each with 100 randomly generated virtual waypoints which would make up the simulated user's virtual path. The same 100 virtual paths were used for each permutation. Each waypoint was generated at a random distance from the previous waypoint using a uniform distribution between 2 and 6 meters. Similarly, each waypoint was generated at a random rotation from the previous waypoint using a uniform distribution between $-\pi$ and π radians. This methodology was derived from the generation of the "Exploration (small)" virtual paths in [1].

For environments A and B, the simulated user started in the center of the environment facing towards the lower Y boundary. As environment C contained an obstacle in the center, the simulated user started in the center of the lower-left quadrant facing the lower Y boundary. At the start of a trial. the simulated user would turn to face the first waypoint and then walk directly towards it. Upon reaching a waypoint the simulated user would stop, turn to face the next waypoint, and again walk directly towards it. This would continue until the simulated user reached the final waypoint. The simulated user turned at a constant rate of $\frac{\pi}{2}$ radians per second and translated at a constant speed of 1 meter per second. The physical component of the simulated user would be redirected using either S2C or P2R. S2C was implemented as described in [6] with the addition of center-based translation gain [1]. Translation and rotation gains for both algorithms were limited to the detection thresholds determined by Steinicke et al. [19]. The maximum curvature for both algorithms was set to a radius of 7.5m, which is a commonly employed threshold value [1, 6].

A reset was triggered upon intersection with one of the boundaries or one of the obstacles. The simulated user's virtual representation would complete a full rotation and their physical representation would rotate to face a target in the physical world, which was determined by the selected reset strategy.

The simulations were run on a Dell PowerEdge R815 with 4x AMD Opteron 6220 processor and 192GB of RAM. All simulations were run with a fixed framerate of 90 fps. Extensive pilot testing of the simulator was done using no redirected walking and S2C. The results were then informally compared against the results from [1] to verify simulator validity.

6 Experiment 1: Optimal Reset Strategy

6.1 Design

The purpose of Experiment 1 was to determine which of the reset strategies described in section 4 work best when coupled with P2R. To accomplish this, a 2x3 study was constructed for each environment with environment size (10m, 20m) and reset strategy (MR2C, R2G, SFR2G) as independent variables. The dependent variables were the number of resets and the mean virtual distance the simulated user traveled between resets. As previously mentioned, SFR2G has two parameters that can be tuned: the number of steps and the step size. Based on internal pilot testing, the number of steps was set to 5 for 10m environments and 15 for 20m environments. The step size was 0.1m for all cases. Experiment 1 was conducted using the aforementioned simulation framework.

For experiment 1, we had two hypotheses. The first hypothesis is that all three reset strategies will not perform significantly differently for environment A. In the empty environment, the gradient field visualization (see Figure 1) shows that the negative gradient always points approximately towards the center of the environment. Therefore, after a reset, the simulated user will be oriented in roughly the same direction for all three reset strategies. However, for environments B and C, we hypothesized that SFR2G would perform better than both R2G and MR2C.

6.2 Results

A Kolmogorov-Smirnov test for normality was conducted for both dependent variables, and these data were not normally distributed. Because of this, results for Experiment 1 were analyzed using non-parametric techniques and the reported values are medians (Mdn) and inter-quartile ranges (IQR). Results for Experiment 1 were analyzed using a Kruskal-Wallis H-test for both the 10m and the 20m conditions. The Mann-Whitney U test was used for post-hoc multiple comparison tests. A significance value $\alpha = 0.05$ was used for all tests. P-values reported for post-hoc multiple comparison tests were adjusted using the Bonferonni method.

6.2.1 Environment A:

Analysis of the number of resets found no significant difference between reset strategies for the 10m condition ($Mdn = 22.0$, $IQR = 4.0$) or the 20m condition ($Mdn = 5.0$, $IQR = 3.0$). Similarly, for the mean distance traveled between resets, there were no significant differences between strategies in the 10m condition ($Mdn = 17.645$, $IQR = 3.321$) or the 20m condition ($Mdn = 77.778$, $IQR = 42.304$). The lack of significant results is consistent with our first hypothesis. We would not anticipate any differences in reset strategy performance, because Environment A is a completely empty space. However, these results confirm that these algorithms are working as expected.

6.2.2 Environment B:

Analysis of the number of resets for the 10m condition found a significant effect for reset strategy, $H(2) = 14.567$, $p < 0.001$. Post-hoc analysis found fewer resets were encountered using MR2C ($Mdn = 35.0$, $IQR = 5.25$) compared to R2G ($Mdn = 37.0$, $IQR = 7.0$), $U = 3685.0$, $p = 0.002$. Additionally, fewer resets were encountered using SFR2G ($Mdn = 35.0$, $IQR = 5.25$) compared to R2G ($Mdn = 37.0$, $IQR = 7.0$), $U = 3621.5$, $p < 0.001$. However, analysis of the number of resets for the 20m condition found no significant difference between reset strategies ($Mdn = 11.0$, $IQR = 4.0$).

Analysis of the mean virtual distance traveled between resets for the 10m condition found a significant effect for reset strategy, $H(2) = 14.944$, $p < 0.001$. Post-hoc analysis indicated that the distance was greater using MR2C ($Mdn = 11.108$, $IQR = 1.832$) compared to R2G ($Mdn = 10.452$, $IQR = 1.881$), $U = 6352.0$, $p = 0.001$. Additionally, the mean virtual distance traveled between resets was greater using SFR2G ($Mdn = 11.021$, $IQR = 1.482$) compared to R2G ($Mdn = 10.452$, $IQR = 1.881$), $U = 6385.0$, $p = 0.001$. However, there were no significant differences between reset strategies for the 20m condition ($Mdn = 34.639$, $IQR = 14.272$).

6.2.3 Environment C:

Analysis of the number of resets for the 10m condition found a significant effect for reset strategy ($H(2) = 6.017$, $p = 0.049$). However, post-hoc analysis found no further significant results. Additionally, there were no significant difference between reset strategies in the 20m condition ($Mdn = 25.0$, $IQR = 5.0$).

Analysis of the mean virtual distance traveled between resets for the 10m condition found a significant effect for reset strategy, $H(2) = 7.89$, $p = 0.019$. Post-hoc analysis found the mean virtual distance traveled between resets was greater using MR2C ($Mdn = 5.966$, $IQR = 0.800$) compared to R2G ($Mdn = 5.785$, $IQR = 0.563$), $U = 5973.0$, $p = 0.034$. Additionally, the mean virtual distance traveled between resets was greater using SFR2G ($Mdn = 5.790$, $IQR = 0.535$) compared to R2G ($Mdn = 5.785$, $IQR = 0.563$), $U = 6014.0$, $p = 0.026$. There were no significant differences between reset strategies in the 20m condition ($Mdn = 15.386$, $IQR = 2.700$).

6.3 Discussion

The results confirmed our first hypothesis and showed that for environment A, there was no statistically significant differnece in performance between any of the three reset strategies. For environments B and C, SFR2G performed better than R2G. These results are consistent with our second hypothesis. However, the results for MR2C were better than we expected; it outperformed R2G and was not significantly different from MR2C for environments B and C. These results made sense after further analysis of some of the individual trials. MR2C would always orient the user in the direction of the most open space in the environments that were tested. SFR2G would orient the user in the direction of the most open space but at a much more local level, only looking as far as the number of steps and step size allowed. R2G, by design, would orient the user directly away from the obstacle or boundary that triggered the reset. If there was not much space between this obstacle and another obstacle or boundary then the user would only have to traverse a small amount before running into the second obstacle/boundary. However, given adequate space, the user would get redirected to face the direction of the most open space. It is possible to conceive of more complex environments where SFR2G might outperform MR2C and more extensive evaluations need to be done to properly examine the differences between them. Further systematic evaluation of the step size and number for SFR2G in different physical environment configurations would also be valuable.

In the three environments tested, SFR2G performed no worse than MR2C, and we speculate that it may perform better in more complex environments. Also, our results suggest that in a larger environment with sparser obstacles, reset strategy may be less important compared to smaller environments that are more densely packed with obstacles. However, in general, our results from Experiment 1 indicate that

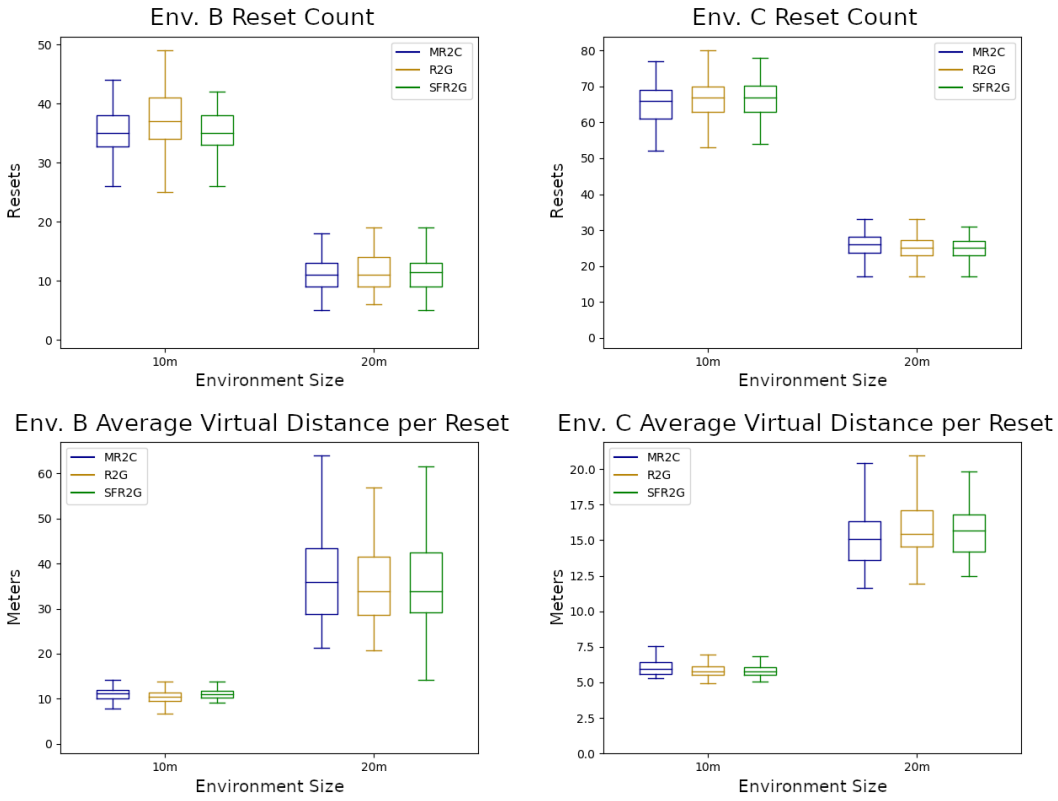

Figure 2: Experiment 1 results for environment B (left), and environment C (right). Boxplots represent the median and IQR. The whiskers represent the data spread not including outliers.

both SFR2G and MR2C are viable reset strategies for non-ideal physical environments.

7 Experiment 2: Comparing S2C and P2R

7.1 Design

The purpose of Experiment 2 was to compare P2R and the previous state-of-the-art reactive RDW algorithm, S2C. To accomplish this, a 2x2 study was conducted with the following independent variables: environment size (10m, 20m) and RDW algorithm (S2C, P2R). Dependent variables included the number of resets and the mean virtual distance the simulated user traveled between resets. Experiment 1 showed that SFR2G and MR2C performed similarly for all of the tested environments. Therefore, we selected MR2C as the reset strategy, because it could readily be applied to both RDW algorithms and provide a fair comparison.

We had two hypotheses for Experiment 2. Similar to the first experiment, we hypothesized that S2C and P2R would have no statistically significant difference in performance for environment A. However, for environments B and C, we hypothesized that P2R would outperform S2C. Upon examining the gradient field for Environment B, shown in figure 1, it can be seen that S2C and P2R will both try to steer the user in roughly the same area, although S2C will attempt to steer the user closer to the obstacle than P2R. Environment C could be considered the worst case scenario for S2C, because the algorithm will be actively attempting to steer the user into the obstacle. Therefore, we would expect a greater difference in performance for this scenario.

7.2 Results

For both dependent variables, a Kolmogorov-Smirnov test for normality was conducted and none of the data was shown to be normally distributed. Because of this, results for Experiment 2 were analyzed using non-parametric techniques and the reported values are medians (Mdn) and inter-quartile ranges (IQR). Results for Experiment 2 were analyzed using the Mann-Whitney U test for pairwise comparisons. A significance value $\alpha = 0.05$ was used for all tests, and p-values reported were adjusted using the Bonferonni method.

7.2.1 Environment A:

Analysis of the number of resets found no significant difference between reset strategies for the 10m condition ($Mdn = 22.0$, $IQR = 4.0$) or the the 20m condition ($Mdn = 5.0$, $IQR = 3.0$). The mean distance traveled between resets was also not significantly different for the 10m condition ($Mdn = 17.784$, $IQR = 3.272$) or the 20m condition ($Mdn = 77.135$, $IQR = 40.523$). Similar to Experiment 1, these results are consistent with our hypothesis, because this was an empty convex environment without obstacles.

7.2.2 Environment B:

Analysis of the number of resets for the 10m condition showed that significantly fewer resets were encountered using P2R ($Mdn = 35.0$, $IQR = 5.25$) compared to S2C ($Mdn = 43.0$, $IQR = 7.0$), $U = 8925.5$, $p < 0.001$. This effect was also observed in the 20m condition, with P2R ($Mdn = 11.0$, $IQR = 4.0$) outperforming S2C ($Mdn = 18.0$, $IQR = 7.0$), $U = 9077.0$, $p < 0.001$.

Analysis of the mean virtual distance traveled between resets also revealed significant effects. In the 10m condition, the simulated user traveled a greater distance using P2R ($Mdn = 11.108$, $IQR = 1.832$) compared to S2C ($Mdn = 9.135$, $IQR = 1.353$), $U = 1017.0$, $p < 0.001$. This effect was also observed in the 20m condition, with P2R ($Mdn = 35.876$, $IQR = 14.742$) outperforming S2C ($Mdn = 21.430$, $IQR = 8.625$), $U = 875.0$, $p < 0.001$.

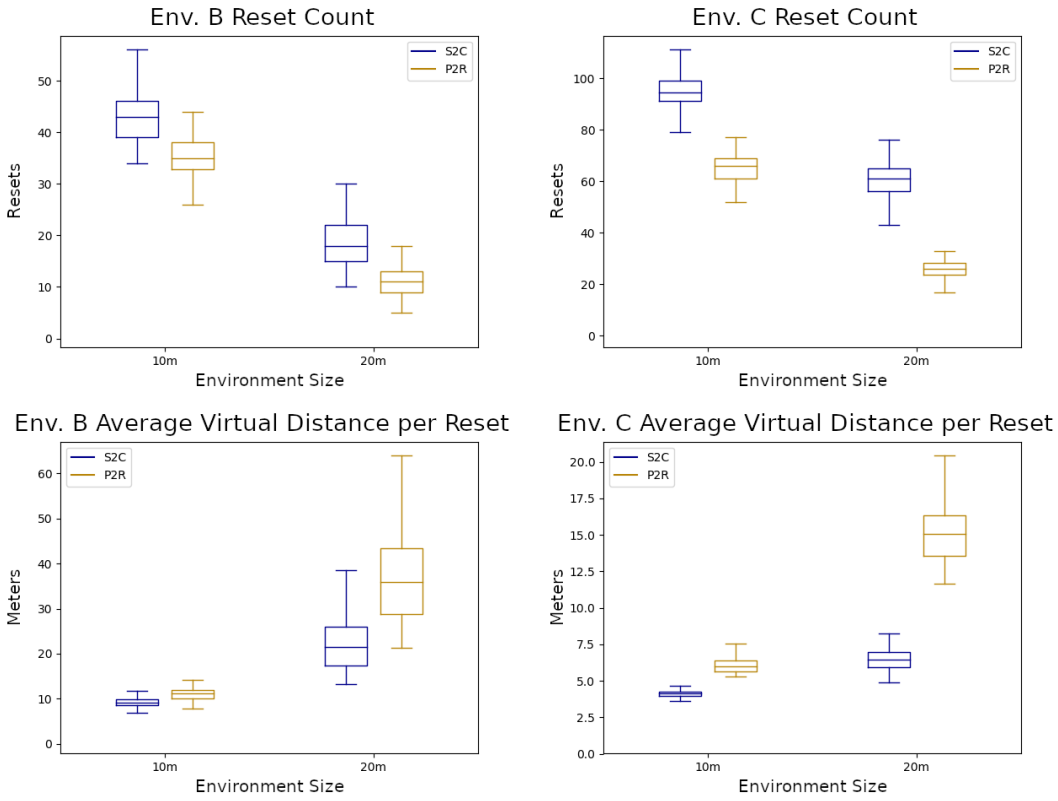

Figure 3: Experiment 2 results for environment B (left) and environment C (right). Boxplots represent the median and IQR. The whiskers represent the data spread not including outliers.

7.2.3 Environment C:

Analysis of the number of resets for the 10m condition showed that significantly fewer resets were encountered using P2R ($Mdn = 66.0$, $IQR = 8.0$) compared to S2C ($Mdn = 94.5$, $IQR = 8.0$), $U = 9989.5$, $p < 0.001$. This effect was also observed in the 20m condition, with P2R ($Mdn = 26.0$, $IQR = 4.5$) outperforming S2C ($Mdn = 61.0$, $IQR = 9.0$), $U = 10000.0$, $p < 0.001$.

Analysis of the mean virtual distance traveled between resets also revealed significant effects. In the 10m condition, the simulated user traveled a greater distance using P2R ($Mdn = 5.966$, $IQR = 0.800$) compared to S2C ($Mdn = 4.129$, $IQR = 0.291$), $U = 0.0$, $p < 0.001$. This effect was also observed in the 20m condition, with P2R ($Mdn = 15.057$, $IQR = 2.739$) outperforming S2C ($Mdn = 6.431$, $IQR = 1.075$), $U = 0.0$, $p < 0.001$.

7.3 Discussion

The performance difference between P2R and S2C in non-ideal environments (B and C) is the most important result reported in this paper. In the most ideal scenario for S2C, both algorithms exhibited similar performance. However, P2R was more robust to non-ideal environments, and was approximately twice as better in some configurations. In general, we can expect that P2R will either perform as well or better than S2C when obstacles are added to the physical environment. As the implementation and computation complexity are not much greater than S2C, P2R therefore appears to be a generally superior reactive algorithm.

As expected, adding an obstacle to the environment greatly reduced redirected walking performance. However, it should be noted that the obstacle used in this experiment was quite large (4x4m in the 10m condition and 8x8m in the 20m condition). Nevertheless, these results reinforce the importance of future research that can address the challenges imposed by non-ideal physical environments.

8 Future Work and Conclusion

There are additional evaluations that can be done to build upon the work presented in this paper. A more thorough examination of new reset strategies that can utilize a potential function is a promising area for future work. Additionally, evaluations comparing P2R with the preliminary algorithm designs proposed in [3] would be valuable.

One of the three limitations of ideal environments that was not addressed in this work was the assumption that the physical environment is static. We believe that P2R can easily be used to solve the most basic form of this problem. As P2R calculates its potential function in a real-time and continuous manner, dynamic obstacles that move through the environment should also repel the user.

In the future, P2R can be extended to alleviate several current limitations of redirected walking. The current form of P2R has a two-dimensional domain for the potential function, but by increasing the number of dimensions in the function's domain, more interesting information can be encoded and acted upon. For example, if the user's virtual information is included than we believe that P2R can be made to behave as a predictive algorithm. Also, by adding another user's information to the domain than P2R could be extended to accommodate multiple users.

In this work, we presented a novel reactive algorithm for redirected walking that does not assume an ideal physical environment. Furthermore, three new reset strategies for environments containing obstacles were proposed and compared. Further evaluations showed that P2R performed as well or better than S2C, the reactive algorithm previously considered to be state-of-the-art, in the environments tested.

Acknowledgments

This research was supported by a Google VR Research Award.

REFERENCES

[1] M. Azmandian, T. Grechkin, M. T. Bolas, and E. A. Suma. Physical space requirements for redirected walking: How size and shape affect performance. In *ICAT-EGVE*, pp. 93–100, 2015.

[2] M. Azmandian, T. Grechkin, and E. Suma Rosenberg. An evaluation of strategies for two user redirected walking in shared physical spaces. In *IEEE Virtual Reality*. IEEE, 2017.

[3] H. Chen, S. Chen, and E. S. Rosenberg. Redirected walking strategies in irregularly shaped and dynamic physical environments. In *IEEE Virtual Reality*, 2018.

[4] T. Grechkin, J. Thomas, M. Azmandian, M. Bolas, and E. Suma. Revisiting detection thresholds for redirected walking: combining translation and curvature gains. In *ACM Symposium on Applied Perception*, pp. 113–120. ACM, 2016.

[5] C. Hirt, M. Zank, and A. Kunz. Geometry extraction for ad hoc redirected walking using a slam device. In *International Conference on Augmented Reality, Virtual Reality and Computer Graphics*, pp. 35–53. Springer, 2018.

[6] E. Hodgson and E. Bachmann. Comparing four approaches to generalized redirected walking: Simulation and live user data. *IEEE Transactions on Visualization and Computer Graphics*, 19(4):634–643, 2013.

[7] C. Hutton, S. Ziccardi, J. Medina, and E. Suma Rosenberg. Individualized calibration of rotation gain thresholds for redirected walking. In *ICAT-EGVE*, 2018.

[8] O. Khatib. Real-time obstacle avoidance for manipulators and mobile robots. In *Autonomous robot vehicles*, pp. 396–404. Springer, 1986.

[9] E. Langbehn, F. Steinicke, M. Lappe, G. F. Welch, and G. Bruder. In the blink of an eye: leveraging blink-induced suppression for imperceptible position and orientation redirection in virtual reality. *ACM Transactions on Graphics*, 37(4):66, 2018.

[10] T. Nescher, Y.-Y. Huang, and A. Kunz. Planning redirection techniques for optimal free walking experience using model predictive control. In *IEEE Symposium on 3D User Interfaces*, pp. 111–118. IEEE, 2014.

[11] C. T. Neth, J. L. Souman, D. Engel, U. Kloos, H. H. Bulthoff, and B. J. Mohler. Velocity-dependent dynamic curvature gain for redirected walking. *IEEE Transactions on Visualization and Computer Graphics*, 18(7):1041–1052, 2012.

[12] A. Nguyen, M. Inhelder, and A. Kunz. Discrete rotation during eyeblink. In *International Conference on Augmented Reality, Virtual Reality and Computer Graphics*, pp. 183–189. Springer, 2018.

[13] N. C. Nilsson, T. Peck, G. Bruder, E. Hodgson, S. Serafin, M. Whitton, F. Steinicke, and E. S. Rosenberg. 15 years of research on redirected walking in immersive virtual environments. *IEEE Computer Graphics and Applications*, 38(2):44–56, 2018.

[14] S. Razzaque, Z. Kohn, and M. C. Whitton. Redirected walking. In *EUROGRAPHICS*, vol. 9, pp. 105–106. Citeseer, 2001.

[15] R. A. Ruddle. The effect of translational and rotational body-based information on navigation. In *Human walking in virtual environments*, pp. 99–112. Springer, 2013.

[16] A. L. Simeone, I. Mavridou, and W. Powell. Altering user movement behaviour in virtual environments. *IEEE Transactions on Visualization and Computer Graphics*, 23(4):1312–1321, 2017.

[17] A. L. Simeone, E. Velloso, and H. Gellersen. Substitutional reality: Using the physical environment to design virtual reality experiences. In *ACM Conference on Human Factors in Computing Systems*, pp. 3307–3316. ACM, 2015.

[18] M. Sra, S. Garrido-Jurado, C. Schmandt, and P. Maes. Procedurally generated virtual reality from 3d reconstructed physical space. In *ACM Virtual Reality Software and Technology*, pp. 191–200. ACM, 2016.

[19] F. Steinicke, G. Bruder, J. Jerald, H. Frenz, and M. Lappe. Estimation of detection thresholds for redirected walking techniques. *IEEE Transactions on Visualization and Computer Graphics*, 16(1):17–27, 2010.

[20] Q. Sun, A. Patney, L.-Y. Wei, O. Shapira, J. Lu, P. Asente, S. Zhu, M. McGuire, D. Luebke, and A. Kaufman. Towards virtual reality infinite walking: dynamic saccadic redirection. *ACM Transactions on Graphics*, 37(4):67, 2018.

[21] M. Usoh, K. Arthur, M. C. Whitton, R. Bastos, A. Steed, M. Slater, and F. P. Brooks Jr. Walking¿ walking-in-place¿ flying, in virtual environments. In *ACM SIGGRAPH*, pp. 359–364. ACM Press/Addison-Wesley Publishing Co., 1999.

[22] M. A. Zmuda, J. L. Wonser, E. R. Bachmann, and E. Hodgson. Optimizing constrained-environment redirected walking instructions using search techniques. *IEEE Transactions on Visualization and Computer Graphics*, 19(11):1872–1884, 2013.