Superflick: a Natural and Efficient Technique for Long-Distance Object Placement on Digital Tables

Adrian Reetz, Carl Gutwin, Tadeusz Stach, Miguel Nacenta, and Sriram Subramanian

University of Saskatchewan
110 Science Place, Saskatoon, Canada, S7N 5C9
adrian.reetz, carl.gutwin, tad.stach, miguel.nacenta, sriram.subramanian @usask.ca

Abstract

Moving objects past arms' reach is a common action in both real-world and digital tabletops. In the real world, the most common way to accomplish this task is by throwing or sliding the object across the table. Sliding is natural, easy to do, and fast: however, in digital tabletops, few existing techniques for long-distance movement bear any resemblance to these real-world motions. We have designed and evaluated two tabletop interaction techniques that closely mimic the action of sliding an object across the table. *Flick* is an open-loop technique that is extremely fast. *Superflick* is based on Flick, but adds a correction step to improve accuracy for small targets. We carried out two user studies to compare these techniques to a fast and accurate proxy-based technique, the radar view. In the first study, we found that Flick is significantly faster than the radar for large targets, but is inaccurate for small targets. In the second study, we found no differences between Superflick and radar for either time or accuracy. Given the simplicity and learnability of flicking, our results suggest that throwing-based techniques have promise for improving the usability of digital tables.

CR Categories: H5.2 [Information interfaces and presentation]: User Interfaces. - Graphical user interfaces.

Keywords: Tabletop workspaces, tabletop interaction techniques, gesture, pen input, radar views.

1 Introduction

Moving objects across a large work surface is a common action in both real-world and digital tabletops. In these tasks people must select and transfer an object to a location that is beyond arms' reach. Real-world examples of this type of action include dealing cards, pushing books across a desk, or sliding tools across the table to another person.

Several techniques have been proposed and studied for improving the efficiency of these long-distance movements. Most of the techniques are based on one of three principles: cursor extensions, such as pantograph-style techniques like Push-and-Throw [7]; long-distance pointing techniques, such as TractorBeam [10]; and proxy techniques that bring distant locations closer to the user, such as Drag-and-Pop [2] or radar views [9].

Even though these techniques are effective, they often add complexity to the tabletop interaction with invocation gestures and mode switches. Furthermore, none of the techniques resemble actions on real world tabletops; and in particular, none of them mimic the way that most people would choose to move objects – by sliding them across the table. Even though some techniques use 'throw' in their names (e.g., Push-and-Throw [7]), they do not involve the basic idea of imparting a velocity and direction to an object in a single quick motion. In a walk-up-and-use tabletop system, we believe that these techniques are problematic in that they require training and may be difficult for infrequent users to remember. In contrast, real throwing-based techniques are easily learned and remembered, use the same basic motions for both local and distant movement (since throwing is just an extension of local placing), and allow other hand-based interactions (such as rotation) to be carried out at the same time as the movement.

Throwing offers another potential benefit – it is based on open-loop rather than closed-loop interaction. Closed-loop techniques like Pantograph require that the user continuously adjust their control movements based on visual feedback about the object's location. Real-world throwing and sliding, in contrast, is open-loop: once the object leaves the person's hand, there is no more control that can be exerted on the object. Open-loop techniques present a tradeoff: they are fast, since the thrower can turn their attention elsewhere as soon as the object leaves their hand, but they require practice in order to achieve accuracy.

In this paper, we design and evaluate sliding techniques for digital tabletops. We were interested in preserving three main principles from real-world throwing:

- *Natural*. The idea of sliding objects across a table is easy to understand and requires no instruction;
- *Lightweight*. Sliding requires little effort and is a natural extension of normal drag-and-drop actions;
- *Fast*. The open-loop nature of sliding means that the technique is extremely efficient, since the action finishes with the initial movement.

Our techniques are called Flick and Superflick. Flick uses a simple stroke on the table surface to slide an object, mimicking the action used to send physical objects across a table. The main benefits of Flick are that it is extremely lightweight and extremely fast; its disadvantage is that it is inaccurate for small targets. Superflick is designed to improve Flick's accuracy. Superflick adds an optional correction phase to Flick – if an initial flick is off-target, the user can immediately put their pen back down on the table and do a 'remote drag-and-drop' to place the object on the target. Since the correction step is only required in cases where the initial flick is off-target, users can reduce their use of the correction as they become more experienced.

We carried out two studies to compare Flick and Superflick with the radar view, a fast and accurate proxy technique [8]. Our results show that for large targets, such as those used when passing objects to other people around the table, Flick is a clear winner: it is accurate enough, and far faster than the radar. For smaller targets, Superflick and Radar are similar in both time and accuracy. Our results suggest that throwing-based interaction techniques – which are already lightweight and easy to remember – are also efficient enough to be used in digital tabletops.

2 Related Work

The idea of integrating desktop computing with physical desks and with the documents commonly found in a workstation has been studied for some time. Wellner's [18] early work attempted to bring physical and digital elements of an office desk closer together through the use of computer vision and projected displays. Other research systems in tabletop collaboration have revealed the potential for effective work and collaboration [4,14,15,16]. Here we review previously-proposed methods for moving and placing artifacts on digital tabletops. Our review focuses on techniques that use direct pointing with a stylus or finger, rather than relative pointing with a traditional mouse (e.g., [1]).

Direct Action. Direct techniques require contact at the initial and final point of interaction. One of the original techniques is Rekimoto's Pick-and-Drop [11] which is an extension of the traditional Drag-and-Drop common in desktop computing. In this implementation, a document can be 'picked up,' by tapping it with a pen, and 'dropped' at another location by tapping the screen once again. These approaches work well, but become difficult on large display surfaces where targets are out of reach.

Cursor Extension. Other interaction techniques used for large displays are the Drag-and-Throw and Push-and-Throw (or Pantograph) methods [7]. The Drag-and-Throw uses a slingshot metaphor where the pen is moved backwards over an object and then released, whereas in the Pantograph technique the pen is moved in the direction of the intended target then released. The distance the object will travel is linearly determined with Push-and-Throw, while the distance is best fit in Drag-and-Throw. Both of these interactions attempt to extend the influence of the user by amplifying their current reach.

Long-Distance Pointing. Parker et al. [10] propose the TractorBeam approach for tabletop interaction. The TractorBeam allows for remote pointing at distant objects on a tabletop, while also supporting touch interaction for objects closer to the user. The initial study found that touching was faster than pointing for small distant targets. Hyperdrag [12] attempts to create a workspace where digital items can be moved freely between displays. With the Hyperdrag technique, a user is able to manipulate documents on any display using their mouse. The Missile Mouse proposed by Robertson et al. [13] attempts to facilitate more rapid pointing with a cursor on a large display. The Missile Mouse technique allows a user to launch a cursor across the screen using a mouse gesture, and stop the cursor by gesturing a second time. A 'wire-guided-missile' approach is also presented, which allows a user to control the path of a launched cursor with mouse movements.

Proxy Techniques. Several techniques work by bringing proxies of potential targets into arms' reach. Drag-and-Pop brings targets that are in the direction of travel closer to the position of the pen [2]. Studies show considerable improvements for Drag-and-Pop (and a related technique, Drag-and-Pick) when targeting in large-display scenarios. The Vacuum [3] is another similar technique that allows users to specify exactly which distant objects should be brought closer.

Radar techniques. Although technically a proxy technique, radar views differ from techniques like Drag-and-Pop in that all objects in the workspace are brought closer using the idea of a workspace miniature. Interaction using radar views was proposed by Swaminathan and Sato [19], in which the 'dollhouse metaphor' is a miniature representation of the larger display. Recently, Biehl et al. [3] developed ARIS, which provides a map of a multi-display environment. Radar views have been shown to be efficient for long-distance movement [9]. One issue with the Radar, however, is that a mode switch is normally required to activate the miniature. This switch can add to the completion time and requires the user to understand the transition to Radar mode [14].

Throwing. Few techniques actually make use of real world throwing motions, although some do use the idea of throwing as the basis for the interaction. For example, Geiβler's throw technique [6] requires the user to make a short stroke over a document, in the direction opposite of the intended target, followed by a long stroke in the direction of the target. The longer the short stroke is, the further the document will travel. Similarly, Wu et al. [21] describe a 'flick and catch' technique, in which an object is 'thrown' once it is dragged at a certain speed (thus it does not use a velocity input model). Finally, Scott et al. [15] extend a rotation and translation technique to include a flicking action for passing and moving items on a tabletop; however the technique is not studied in detail.

3 DESIGNING BASIC FLICK

In the real world, flicking and sliding actions depend on several variables, including the weight of the object, the force that is applied to move the object, the direction of the force, and the friction of both the object and the surface. These factors determine an initial direction and velocity, and the final position of the object can easily be calculated using a physics model.

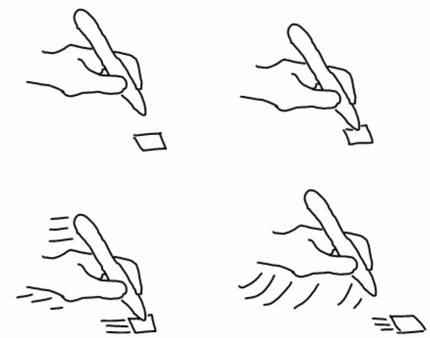

Figure 1. Stages of a flick

We experimented with several models that had varying degrees of fidelity to real-world physics. We found that it was easy to come up with a model that seemed close to people's expectations, but difficult to find a model that allowed people to be as accurate as they could when sliding real objects. The main problem was that timestamps on input events are not exact: although the time is high-resolution (we were able to record 50 samples per second), it did not correspond exactly to the moment that the sample was received. Therefore the recorded data was noisy, making accurate velocity calculations difficult (Figure 2). We tested Gaussian filtering and frequency filtering with Fourier transform, but the most consistent results were found with a first-degree Least Square Method regression. We use the last ten samples to calculate both velocity and direction.

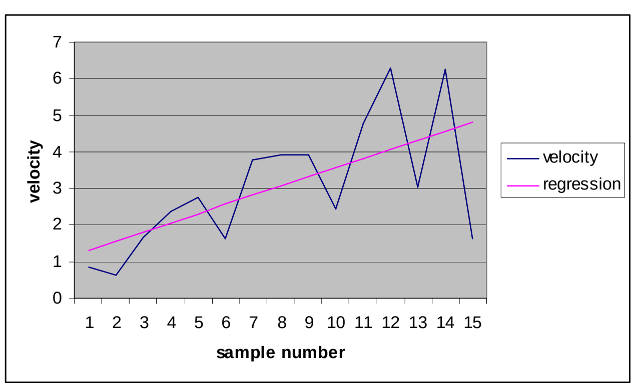

Figure 2. Gesture velocity at different sample numbers

4 Pilot Study: Flick vs. Radar

We carried out a pilot study to compare Flick with Radar. Although the study involved only a small number of participants, the results showed clear differences between the two techniques.

4.1 Pilot Apparatus and Participants

A custom system was built in C++ for the experiment, and was installed in a top-projected tabletop system (Figure 1). The table was 125x89 cm, and the projector had a resolution of 1024x768 pixels. Participants used a Wacom tablet (21x15 cm) as the input device (note however that the techniques can work with any direct-input device).

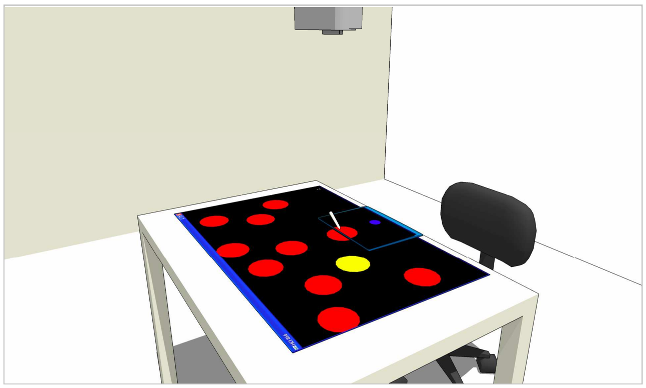

Figure 3. Experimental setup.

Four participants (3 male and 1 female) were recruited from a local university. Participants ranged in age from 18 to 21 years and averaged 20.5 years. All were familiar with mouse-and-windows applications (i.e., more than 8 hours per week); however, none had previous experience with a digital tabletop.

4.2 Pilot: Design and Experimental Conditions

The study used a 2x4 repeated-measures factorial design. The factors were technique (Flick or Radar) and target size (small, medium, large, or infinite).

Radar. The radar view is a proxy technique that displays a working miniature of the entire workspace. In our implementation, the radar view appeared as soon as the participant touched the digital object, and the full-size object was replaced by its miniature equivalent (see Figure 4). The participant then dragged the pen to the target (in the radar view) and lifted the pen to complete the trial (see video figure at http://hci.usask.ca).

Flick. We implemented the pure open-loop flick technique as described above. Participants put the pen down on the digital object, dragged the pen towards the target, and released the pen to throw the object. Once this initial gesture was complete, there were no further control actions possible.

There were four target sizes: small (17cm / 140 pixels), medium (24cm / 200 pixels), large (30cm / 240 pixels), and infinite (the target was 30cm / 240 pixels wide, and touched the edge of the table, meaning that it was infinitely deep; see Figure 5). Infinite targets were included to test the real-world situation of giving objects to another person seated around the table where moving objects stop at the table boundary. There were also three target locations, as shown in Figures 5 and 6: left, top, and right.

Participants were asked to carry out a series of object-movement trials using first the Radar, and then Flick. Participants completed 50 training trials in each interface, then 100 testing trials.

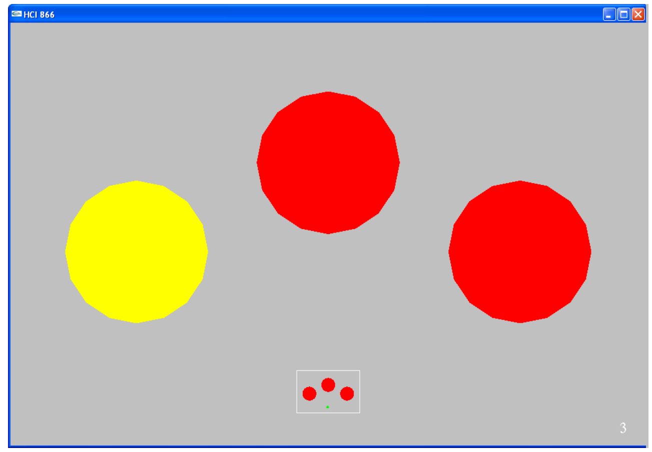

Figure 4. Radar interface, as displayed immediately after touching the digital object.

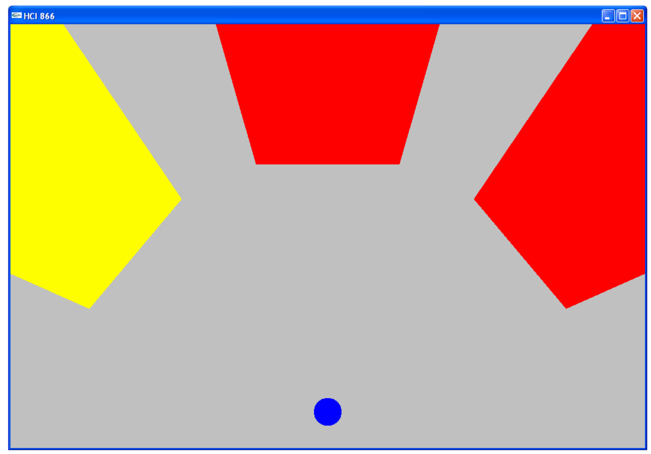

Figure 5. Infinite targets; yellow target (left) is the current target. Object to be flicked is in blue at bottom.

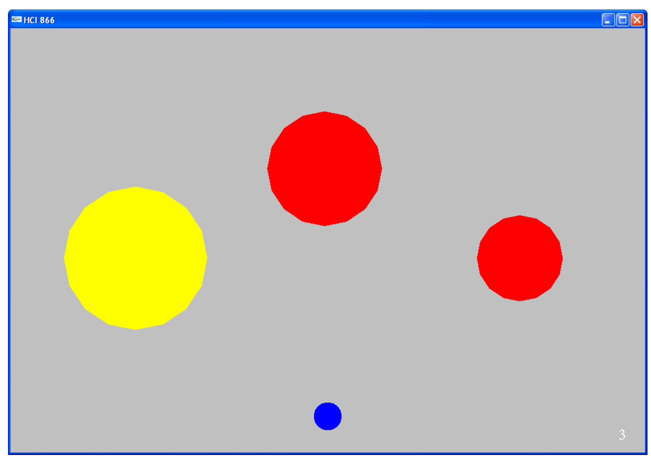

Figure 6. Small (left), medium (top), and small (right) targets (all sizes appeared equally in all locations).

4.3 Pilot: Results

Completion time. The overall completion time across both conditions was less than half a second (mean 394ms, s.d. 166ms). Even with only four participants, there were main effects of both technique ($F_{2,6}$=53.42, p<0.001), and target size ($F_{3,9}$=31.30, p<0.001). As shown in Figure 7, completion time for Flick is approximately half the best time for Radar, and larger targets result in faster times than smaller targets.

However, there was also a significant interaction between technique and target size ($F_{6,18}$=18.34, p<0.001). As can be seen in Figure 7, completion time for Flick is almost static across all target sizes, in keeping with the open-loop nature of the technique.

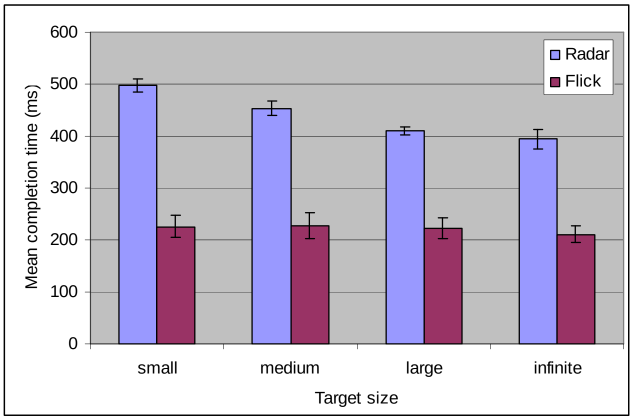

Figure 7. Mean completion time (pilot).

Accuracy. Accuracy was recorded as a simple 'hit' or 'miss' on each target. Overall mean accuracy was 88% (s.d. 19%), but there were again large differences between the conditions. There were significant main effects of both technique ($F_{2,6}$=53.42, p<0.001) and target size ($F_{3,9}$=33.02, p<0.001); and there was a significant interaction between the two factors ($F_{6,18}$=27.45, p<0.001). In this case, however, it is the radar that is invariant across target sizes, whereas accuracy with Flick ranges from about 50% for small targets, to 95% for infinite targets (see Figure 8).

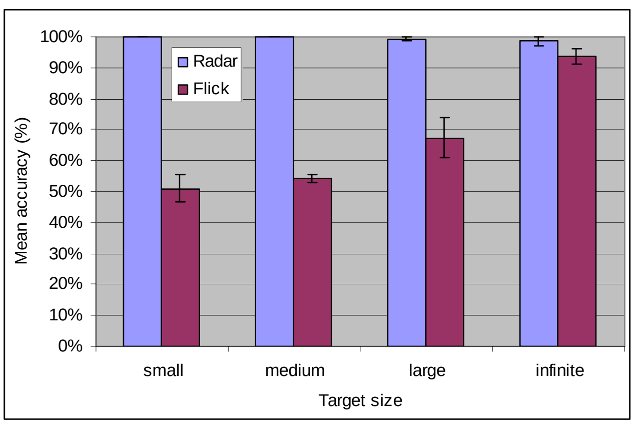

Figure 8. Mean accuracy for all target sizes (pilot).

4.4 Discussion of pilot study results

The pilot study showed an extremely clear time-accuracy tradeoff between the two techniques: Flick is always fast, but accuracy drops with decreasing target size; Radar is always accurate, but completion time increases with decreasing target size.

The design implications of the pilot are also clear: Flick is an excellent technique for targets that touch the edge of the table, such as in the case of passing an object to another person around the table. These types of targets completely overcome the distance inaccuracy of Flick; and in these situations, the technique clearly has a place in the designer's toolbox.

However, if targets are small and accuracy is important, then the Radar view is far superior. Because of this difference for small targets, we decided to redesign Flick to try and improve the technique's accuracy.

5 The Design of Superflick

Superflick adds an optional closed-loop control step to basic Flick. We wanted to keep the speed and simplicity of regular Flick, but allow corrections when the original motion was inaccurate. We therefore enabled 'remote drag-and-drop' on the thrown object: if the user puts their pen back down on the table while the object is still moving, they can adjust the final position by dragging (see Figure 9). It is important to note that the user does **not** have to wait until the throw is finished: the system knows the final position of the object as soon as it is thrown (since the motion is deterministic), and displays the final position as soon as the flick gesture occurs (see video figure at hci.usask.ca). The 'remote drag-and-drop' acts on this final position, not the moving object; this means that the user can correct the position immediately after releasing the object, and that they do not have to guide the object as it moves (as in the 'wire-guided-missile' approach).

Superflick's correction step is optional. If the user hits the target with the initial Flick, no further actions are necessary. When they miss the target (and they can see this as soon as they release the object), they can immediately manipulate the final position using the correction step. In order to allow larger corrections, we use a 1:4 control-to-display ratio in the correction step. The addition of the correction step gives users of Superflick the ability to achieve 100% accuracy.

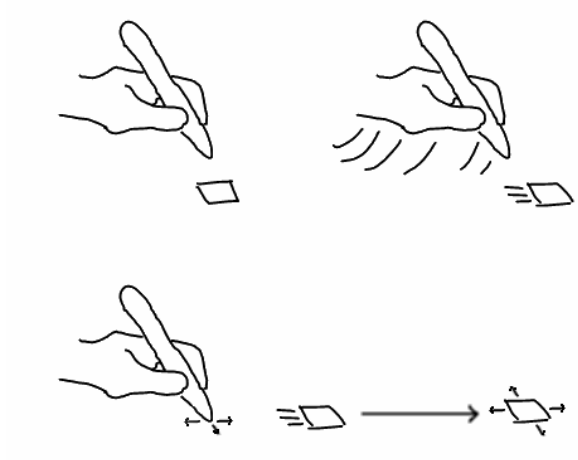

Figure 9. Stages of Superflick technique

6 Comparison Study: Radar, Flick, and Superflick

We carried out an experiment that compared Flick and Superflick with a radar view for a variety of placement tasks. Again, our goal was to determine whether flick-based techniques could approach the efficiency of existing approaches like the radar: flicking has advantages in simplicity and ease of learning, and we wanted to see whether those advantages came at an efficiency cost.

6.1 Apparatus and Participants

The apparatus used in the comparison study was the same as that used in the initial pilot study. Twelve participants (6 men and 6 women) were recruited from a local university. Participants ranged in age from 19 to 26 years (mean 22.1). All were familiar with mouse-based applications (>8 hours/week).

6.2 Design and Experimental Conditions

The study used a within-participants 3x1 factorial design. The single factor was the interface type: Radar, Flick, or Superflick. Although the pilot showed that Flick has accuracy problems, we included it to have a baseline for comparing the performance of Superflick.

Radar. The radar view functioned as described above, but for this study an invocation gesture was added. This required the user to make pen contact outside of the digital object and then drag the pen tip inside the object in order to activate the radar. We added this mode switch after realizing that it would be impossible for radar users to differentiate between long-distance actions and local drag-and-drop actions (see below for further discussion of this decision).

Flick. The flick technique was identical to the method used in the pilot study. Note that for both Flick and Superflick, no invocation gesture is required because both of these techniques are simply extensions of an existing local-movement technique (Drag-and-Drop).

Superflick. The Superflick technique was also implemented as described above. Participants begin with a flick gesture; as soon as the gesture is complete, the object's final location is displayed, and the participant can put the pen back down on the table to move the object (at a 1:4 C:D ratio).

In the comparison study we used only one target size (17 cm / 140 pixels), and we used a different target arrangement than in the pilot. In this study, a set of circles was displayed in random locations (see Figure 10), and the next target was chosen randomly from among these. Trials were timed slightly differently due to differences between the techniques. Since the radar is a closed-loop technique, timing of the trial ended when the pen was released at the end of the object movement. The open-loop nature of the flick techniques required different timing. Since Flick is completed at the end of the flick gesture, we used this for the trial's end time. This is reasonable, since the user can turn their attention to other objects as soon as they release the object (and also since we show the final position of the object immediately upon release). Superflick was timed similarly to Flick in the cases where no correction step was undertaken; in cases where a correction was made, the trial was timed until the end of the correction.

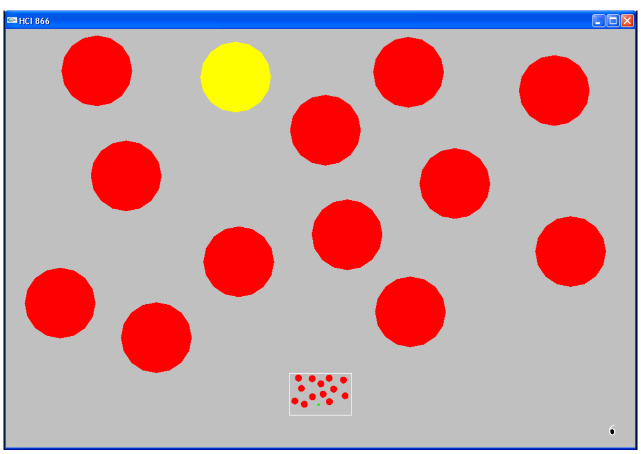

Figure 10. Radar interface, as displayed immediately after dragging the pen into the digital object.

6.3 Procedure

Participants were first introduced to the three interaction techniques. Participants then carried out fifteen blocks of ten trials with each technique (five training blocks, and ten test blocks). At the end of the study, they completed an overall preference survey.

The study system collected time and error data for all trials; in addition, questionnaire data was recorded after the trials. With 12 participants, each carrying out 150 trials with each of the 3 interfaces, the system collected data from a total of 5400 trials.

6.4 Results

Completion Time. Over all techniques, the mean completion time was 791ms (s.d. 309ms). There was a significant main effect of technique ($F_{2,22}$=56.27, p<0.001); as can be seen in Figure 11, Flick was again the fastest technique. T-tests show that Flick is significantly faster than both the other techniques (p<0.001); there was no difference between Superflick and Radar.

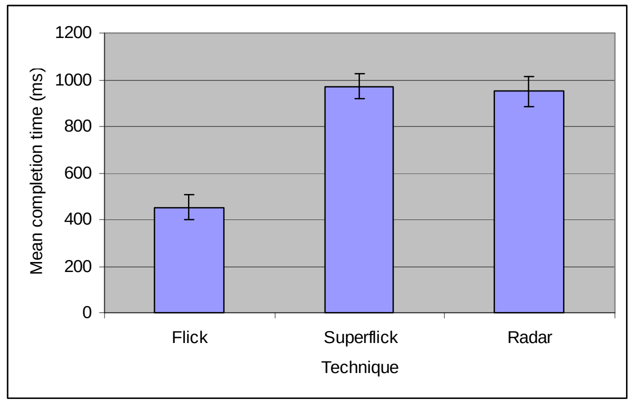

Figure 11. Mean completion times for all tasks; error bars show standard error.

Performance over time. We also carried out a post-hoc analysis using trial block as an additional factor (including training trials as well as testing trials). We found a main effect of block number ($F_{19,209}$=13.83, p<0.001), and a significant interaction between block and technique ($F_{38,418}$=12.44, p<0.001). As can be seen from Figure 12, performance improved with both Radar and Superflick, but did not with Flick (in fact, completion time rose slightly over time for Flick).

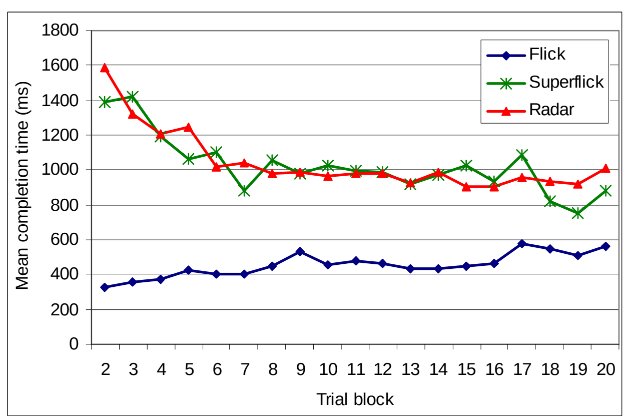

Figure 12. Completion time by trial block (including training in blocks 2-5; block 1 was demonstration).

Accuracy. There was a main effect of Technique ($F_{2,22}$=422.28, p<0.001); as in the pilot study, accuracy rates were again dramatically different between Flick and Radar. Followup t-tests show that Flick is lower than both the other techniques; again, there was no differences between Superflick and Radar.

Accuracy over time. As with completion time, we tested accuracy by trial block (again using practice trials). There was no main effect of block ($F_{19,171}$=1.57, p=0.065), indicating that accuracy did not change throughout the study. There was also no interaction between block and technique ($F_{38,418}$=1.39, p=0.065) (see Figure 14).

Effort and Preference. A post-study questionnaire was given to each participant. Each of the three techniques were given a subjective score for a series of measurements. Figure 15 shows the average rating given by the participants and is scaled from positive to negative. Overall, Radar was the preferred technique, and Flick was seen as frustrating (likely due to its high error rate).

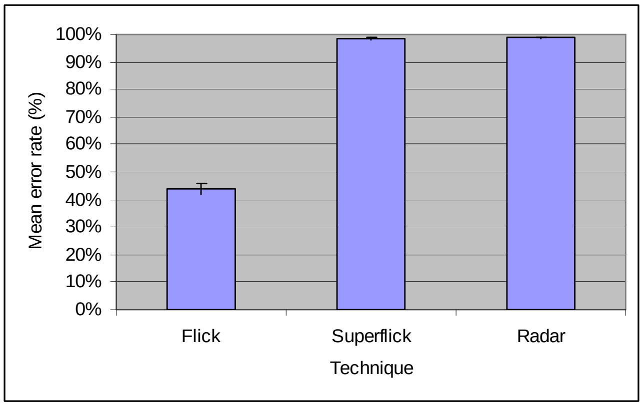

Figure 13. Mean accuracy for all tasks.

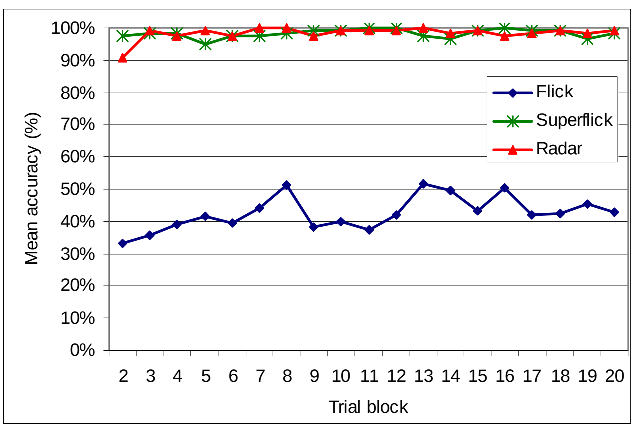

Figure 14. Mean accuracy by trial block (including training in blocks 2-5; block 1 was demonstration).

7 Discussion

Our studies identified the major strengths and weaknesses of each of the three techniques:

- Flick is extremely fast, requiring less than half the time of the other techniques on average. For infinite targets (e.g., other people around a table), Flick is accurate enough for real-world use. For all other target sizes, however, Flick was far less accurate.
- Superflick corrects the accuracy problems of Flick, and requires approximately the same time as Radar (no significant differences were found).
- Radar was reasonably fast (less than one second per trial),, and extremely accurate, on all target sizes. Radar is slower when the visual field is more complex (as in the second study). In addition, Radar was preferred by the participants.

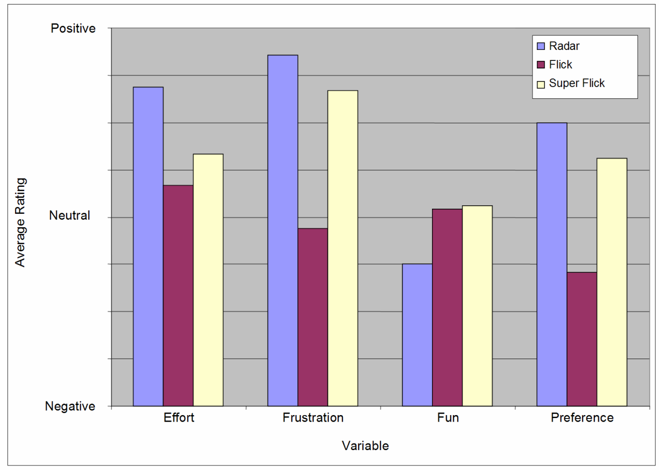

Figure 15. Average user ratings for each technique.

7.1 Explanations of Results

Here we consider explanations for three of our study's results: the overall speed of Flick, the performance of Superflick, and the performance of Radar.

Flick was the fastest technique, and it seems clear that its speed advantage comes from its open-loop design. Flick showed no change in speed throughout the studies; it always took approximately the same amount of time regardless of the size of the target.

Superflick successfully addressed the accuracy problems with Flick, and did so without adding an undue amount of time to the technique. There is a relationship between speed and target size in Superflick, because of the time needed to carry out closed-loop corrections. The performance of Superflick over the long term, therefore, is dependent on the proportion of initial flicks that are successful: more good initial flicks means less time spent in correcting. In future work we will study people's ability to improve their initial accuracy with continued experience.

The radar view proved to be an excellent all-round technique, as has been found before [9]. Even with an invocation gesture, and a visual disconnect between the miniature and the main display, the Radar was fast, easy to learn, and preferred by many of the participants. One question about the radar is that of why it was slower in study two than in the pilot. There are two likely reasons. First, the invocation step, although extremely lightweight, does add some time to the technique. Second, and more importantly, the visual field in the second study used more objects in a more complicated visual layout. In the pilot study, participants did not really need to look at the main tabletop in order to use the radar; they could determine which of the three targets was yellow through peripheral vision, and focus their attention on the radar display. This is an unrealistic situation for many tabletop systems, where there will be multiple objects (e.g., pictures, documents, artifacts, tools) distributed on the display. In the second study, with the more complicated visual scene, we noticed people looking back and forth from the radar to the main display to make sure that they were approaching the correct object (since the target was not highlighted in the radar). This checking action was the main reason for the radar's additional time.

7.2 Issues in the Design of the Techniques

Here we consider issues in the design of our techniques that may have affected our study results. First, we consider the issue of adding the invocation gesture to the radar for the second study.

This was done, as described above, because the radar requires a means for differentiating between local drag-and-drop (without the radar) and long-distance moves (that use the radar). However, we could have assumed that the mode switch would be done for local moves rather than long-distance moves, which would have improved the radar's performance in the study. However, we felt that using the gesture for the long-distance case was more realistic: we felt that users of a real-world system would be more confused if they had to use a gesture before a local drag-and-drop than if they did it to invoke the radar.

In contrast to Radar, Superflick does not require mode switches. On closer examination drag-and-drop and flicking are very similar movements in the real world. The major difference is that people release the object at a certain speed if they want to flick or slide it, and they hold on to the object if they want to drop it. The Flick implementation works for both cases: when dropping an object locally, the user's motion slows to zero while still holding the object, and so the system gives the object an initial velocity zero, which is equivalent to a drop. For both Superflick and drag-and-drop, the system does not need to know the user's intention, because the same formulas can be applied.

Second, the way that we timed trials for Superflick means that the times are very slightly lower than they should be – because we could not time the visual evaluation of the initial flick. That is, in cases where the initial flick is accurate, the user still has to visually evaluate that the object is correctly touching the target, but there was no way for us to measure this visual evaluation time. In cases where the user corrects the location, however, we do get accurate data because the time extends to the end of the correction. We believe that this visual evaluation step occurs extremely quickly; however, we plan to test it with a followup study that asks participants to move as many objects as they can within a set time period.

Third, we decided to animate the process of sliding for all three interaction techniques. Although this is not required for the radar view, we wanted the visual feedback of all three techniques to be similar. The trial timing did not include the animation, however (radar actions were timed only to the release of the pen), so this technique was not disadvantaged by the animation.

7.3 The Techniques in Real-World Applications

Several issues must be considered when applying our results to real-world applications. First, our experimental setup used simplistic circular targets; this may have some impact on the generalizability of the Radar in particular. For example, real icons may become unrecognizable in a radar view because of the greatly reduced resolution, making target recognition difficult.

Second, Flick and Superflick must be used with a table-wide input system, rather than a tablet as was used in our studies. To ensure that this is easily done, we re-implemented the flick techniques with finger-based input (using a Polhemus tracker); no difficulties were encountered in developing this new system.

Third, we tested only long-distance movement. Real tabletop work involves both local and longer-range actions, and the underlying interaction techniques should be able to support both ranges. Radar requires an explicit mode switch to shift from normal Drag-and-Drop movement to long-range movement, whereas Superflick uses the same technique for both ranges. It remains to be seen whether there will be any difficulty in moving items only a short distance with Superflick, or whether there will be mode errors with the Radar.

Fourth, compared with people's sliding performance in the real world, it might seem difficult to achieve both naturalness and accuracy in the same technique. However, and primarily because people can become skilled at sliding in the real world, we believe that both of these goals can be met. Our difficulties with the techniques are mostly technical: Figure 2 illustrates the problem of low sample rate and high noise, one of the major drawbacks of any time sensitive system. We believe that with improved time measurements and sampling rates, we would be able to map the user's input more accurate to our physical model and provide more effective feedback to the user. This also would help users to gain a better understanding of Flick and help them to develop better accuracy over time with this technique.

8 CONCLUSIONS

Of the many techniques that have been developed for moving objects on digital tabletops, very few have been based on the natural sliding actions that are common in the real world. We found this surprising, since real-world sliding is natural, lightweight, and uses the same basic actions for local and distant movements. We designed two techniques that are based on sliding, and tested them to see whether they could be as efficient as existing approaches. Our first technique, Flick, was shown to be extremely fast, but can only realistically be used for large targets. The second technique, Superflick, provides a correction step for cases where the initial flick is off target. A second study showed that Superflick fixes the accuracy problems seen in Flick; no differences between Superflick and Radar were found, although times and accuracies were similar. Since Superflick is easy to learn, does not require a mode switch, and will approach the speed of Flick for large targets, we believe that it should be considered by designers of digital tabletop applications.

In future work, we plan several extensions to, and further studies of, the techniques. As mentioned above, both Radar and Superflick should be investigated in tabletop applications with realistic targets and usage patterns that involve both local and long-distance object movements. Second, we plan to tackle Flick's timing and sampling issues by using a more accurate input system such as an A/D board. This may give us more precise time and coordinate data and thus a more consistent velocity model, that will help to improve the distance accuracy of Flick. Finally, we will look at combining other interactions that are possible on real-world tables, such as rotation, with the flick techniques.

9 ACKNOWLEDGMENTS

This research was supported in part by the Natural Sciences and Engineering Research Council of Canada, and by the NECTAR research network. Our thanks to Bernard Champoux for assistance with figures.

REFERERNCES

[1] Asano, T., Sharlin, E., Kitamura, Y., Takashima, K., Kishino, F. Predictive Interaction using the Delphian Desktop, *Proceedings of UIST 2005*, 133-141.

[2] Baudisch, P., Cutrell, E., Robbins, D., Czerwinski, M., Tandler, P. Bederson, B., and Zierlinger, A., Drag-and-Pop and Drag-and-Pick: Techniques for Accessing Remote Screen Content on Touch- and Pen-operated Systems, *Proceedings of Interact 2003*, 57-64.

[3] Bezerianos, A. and Balakrishnan, R., The vacuum: facilitating the manipulation of distant objects. *Proceedings of CHI 2005*, 361-370.

[4] Biehl, J. and Bailey, B. ARIS: An Interface for Application Relocation in an Interactive Space. In *Proceedings of Graphics Interface 2004*, 107-116.

[5] Deitz, P. and Leigh D., DiamondTouch: A Multi-User Touch Technology, *Proceedings of ACM UIST 2001*, 219-226.

[6] Geiβler, J. Shuffle, throw or take it! Working Efficiently with an Interactive Wall, *Proceedings of CHI 1998*, 265-266.

[7] Hascoët, M. Throwing models for large displays. In *Proceedings of HCI´2003*, British HCI Group, 73-77.

[8] Moyle, M., and Cockburn, A., Analyzing Mouse and Pen Flick Gestures., *Proceedings of the SIGCHI-NZ Symposium On Computer-Human Interaction*, Hamilton, New Zealand, 2002, 266-267.

[9] Nacenta, M., Aliakseyeu, D., Subramanian, S., and Gutwin, C., A Comparison of Techniques for Multi-Display Reaching., *Proceedings of ACM CHI 2005*, 371-380.

[10] Parker, J., Mandryk, R., and Inkpen, K., TractorBeam: Seamless Integration of Local and Remote Pointing for Tabletop Displays, *Proceedings of Graphics Interface 2005*, 33-40.

[11] Rekimoto, J., Pick-and-Drop: A Direct Manipulation Technique for Multiple Computer Environments, *Proceedings of ACM UIST 1997*, 31-39.

[12] Rekimoto, J. and Saitoh, M., Augmented Surfaces: A Spatially Continuous Work Space for Hybrid Computing Environments, *Proceedings of ACM CHI 19'99*, 378-385.

[13] Robertson, G., Czerwinski, M., Baudisch, P., Meyers, B., Robbins, D., Smith, G. and Tan, D., The Large-Display User Experience, *IEEE Computer Graphics and Applications*, 25(4), July/August 2005, 44-51.

[14] Scott, S., Grant, K., and Mandryk, R., System Guidelines for Co-located, Collaborative Work on a Tabletop Display, *Proceedings of ECSCW 2003*, 159-178.

[15] Scott, S., Carpendale, S., and Habelski, S., Storage Bins: Mobile Storage for Collaborative Tabletop Displays. *IEEE Computer Graphics and Application*, 25(4), July/August 2005, 58-65.

[16] Shen, C., Lesh, N., and Vernier, F., Personal Digital Historian: Story Sharing Around the Table. *ACM Interactions*, 10(2), March/April 2003, 15-22.

[17] Shen, C., Vernier, F., Forlines, C., and Ringel, M., DiamondSpin: an Extensible Toolkit for Around-The-Table Interaction, *Proceedings of ACM CHI 2004*, 167-174.

[18] Streitz, N., Geißler, J., Holmer, T., Konomi, S., Müller-Tomfelde, C., Reischl, W., Rexroth, P., Seitz, P., and Steinmetz, R., i-LAND: An interactive Landscape for Creativitiy and Innovation, *Proceedings of ACM CHI 1999*, 120-127.

[19] Swaminathan, K. and Sato, S., Interaction design for large displays. *ACM Interactions* 4(1), 1997, 15-24.

[20] Wellner, P., Interacting with Paper on the DigitalDesk, *Communications of the ACM*, 1993, 87-96.

[21] Wu, M., and Balakrishnan, R., Multi-finger and whole hand gestural interaction techniques for multi-user tabletop displays. *Proceedings of ACM UIST 2003*, 193-202.