Clutch-Free Panning and Integrated Pan-Zoom Control on Touch-Sensitive Surfaces: The CycloStar Approach

Sylvain Malacria **Eric Lecolinet** **Yves Guiard**

Telecom ParisTech – CNRS LTCI UMR 5141
46 rue Barrault, 75013, Paris, France
{sylvain.malacria; eric.lecolinet; yves.guiard}@telecom-paristech.fr

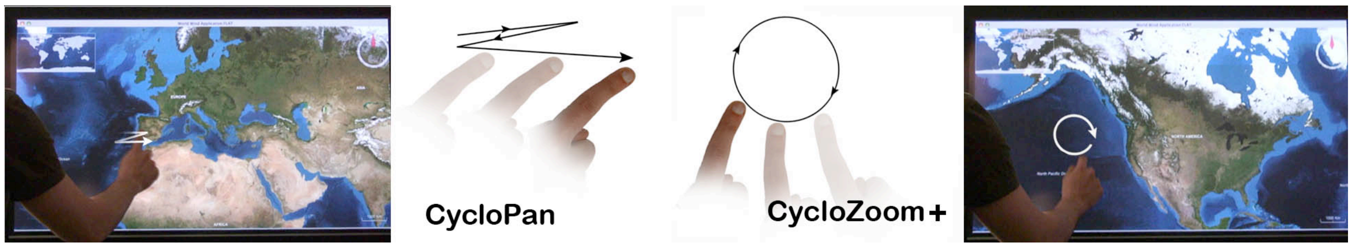

Figure 1: CycloPan (left), CycloZoom+ (right), along with the schematic form of their controlling gestures.

ABSTRACT
This paper introduces two novel navigation techniques, CycloPan, for clutch-free 2D panning and browsing, and CycloZoom+, for integrated 2D panning and zooming. These techniques instantiate a more generic concept which we call Cyclo* (CycloStar). The basic idea is that users can exert closed-loop control over several continuous variables by voluntarily modulating the parameters of a sustained oscillation. Touch-sensitive surfaces tend to offer impoverished input resources. Cyclo* techniques seem particularly promising on these surfaces because oscillations have multiple geometrical and kinematic parameters many of which may be used as controls. While CycloPan and CycloZoom+ are compatible with each other and with much of the state of the art, our experimental evaluations suggest that these two novel techniques outperform flicking and rubbing techniques.

ACM Classification Keywords
H5.2 [Information interfaces and presentation]: User Interfaces. - Graphical user interfaces.

Author Keywords
Input techniques, touch screens, touchpads, oscillatory motion, elliptic gestures, panning, zooming, multi-scale navigation.

General Terms
Design**,** Human Factors.

INTRODUCTION
One conspicuous feature of the current evolution of interactive devices is the spread of touch-sensitive surfaces. Not only are small touch screens increasingly used in mobile devices, especially smartphones, but with the popularization of interactive tabletops and whiteboards, also enlarged interactive surfaces are becoming more and more common. These devices often miss a hardware keyboard and they generally offer few physical buttons. In the case of small devices such as smartphones this trend results from an attempt to maximize the screen real estate while keeping a small form factor. Much the same problem arises on large surfaces like interactive whiteboards or tabletops but for the opposite reason: the large size of the device makes it difficult or impossible to reach hard buttons located around the device while touching the surface at the desired location.

Users thus miss common input resources like the physical keyboard for entering text, hotkeys, and dedicated keys and mouse buttons for specifying states [6], for opening a context menu, etc. And there is of course no equivalent of the mouse-wheel. Interaction must therefore heavily rely on the capabilities of the touch-screen itself. Gesture analysis seems to have a promising potential for enriching the input vocabulary and considerable efforts in HCI research have been made in this direction over the last decade. Cyclo* ("CycloStar"), the approach presented in this paper, aims at generalizing techniques using oscillatory movements for controlling continuous variables.

The Cyclo* concept is generic, meaning that many different interaction techniques can be instantiated from it, depending on task context. This paper presents two such instances. *CycloPan* provides an efficient way for panning documents that avoids clutching. *CycloZoom+* allows to simultaneously zoom and pan documents in a smooth way. CycloPan and CycloZoom+ are compatible with each other as well as with conventional techniques such as dragging and flicking. Users can trigger the technique they need just by performing the appropriate gesture, no explicit mode switch being required. Finally, if Cyclo* techniques are especially useful for small and large interactive surfaces, they can also help with conventional input devices such as touchpads, as will be shown in the description of Experiment 1.

RELATED WORK

Multi-Touch Gestures

Multi-touch gestures provide ways of achieving some of the operations described above, but they suffer some limitations of their own. In the case of mobile interaction, that resource is out of reach when a single hand has to both hold and operate the device. There are numerous cases in real-life mobility conditions where one-handed interaction is needed simply because only one hand is free [23]. Another obvious problem is that multi-touch interaction requires appropriate hardware not yet available in many existing devices, notably most interactive whiteboards. Moreover, efficient multi-touch technology for large surfaces is still expensive and often bulky (especially when relying on video analysis).

Gestures as Spatial Paths

Gestures conceal rich resources that can serve as substitutes for some of the abovementioned missing resources. Even if it is analyzed as a geometrical path, without any reference to the time dimension, a gesture made on a sensitive screen specifies a certain shape and thus conveys more information than traditional pointing, which only specifies the location of a point in the plane.

The geometrical analysis of gestures has been used in a number a techniques for selecting discrete items or for setting continuous values. Various kinds of gestures have been considered for discrete selection, e.g., the well-known Marking menus [19], which use straight radial gestures from the activation point, and Flower menus [3], which consider path curvature too. Rolling gestures have been used in specific contexts such as text input in Rollpad [22], or to differentiate pointer states in SimPress [4]. MicroRolls [25] have been proposed as general-purpose gestures that extend the gestural vocabulary for touch screens.

Other techniques based on gesture path analysis allow users to set a value in a given interval of a continuous variable. Applied to Marking menus, this idea gave birth to Flow and Control menus [15, 24]. To set a continuous parameter the system considers the distance covered by the cursor from an activation point. With the Virtual Scroll Ring [20], Radial Scroll [26], Curve Dial [27] and Chiralmotion [2] circular oscillatory motions are exploited for scrolling documents in one direction. These techniques either consider the distance traveled along the circumference of a circle [20] or rely on the Vernier effect [26, 27], the angular variation controlling scrolling speed by adjusting the radius [10]. Finally, gestural paths combined to spatial location allow to control the 6 degrees of freedom (DoF) of a camera [29].

Kinematics: Gestures as Trajectories in Space-Time

With kinematic analysis, which considers not simply geometrical paths but space-time trajectories, yet more information can be extracted from the input flow on a sensitive surface. For any particular path traced by the finger on a tactile surface, the user may have produced an infinite number of different velocity profiles. Kinematic analysis thus exploits a larger proportion of the information conveyed by gestures than does geometric analysis [28].

A well-know instance of a kinematics-based technique is *flicking,* or swiping, a scrolling technique now available in widely-spread commercial systems like Google Earth and the iPhone. A flick and a drag are easy to distinguish kinematically: at release finger velocity and acceleration are typically positive for a flick and close to zero for a drag [1].

Oscillatory Gestures

Of special interest, within the class of input techniques that resort to the kinematic analysis of motion, are those which use gestures of the *cyclical* category. It is known that it is quite easy for humans to settle and sustain stable oscillations with any part of the body, in particular with the hand [18]. Whether the cyclical gesture traces an ellipse, a circle or a simple line segment, we have the interesting property that the parameters of the oscillatory motion, which a user can modulate at every single instant, can be used as controls.

In Motion Pointing [11] each target of a large set had one pixel (called the driver) oscillate along a small ellipse with a unique combination of geometric and kinematic parameters (size, eccentricity, and tilt of the ellipse; frequency and direction). Users select a particular target by simply mimicking its specific ellipsoidal oscillation, so that this technique makes it possible to point without a pointer [28]. Motion Pointing, based on the match between the oscillatory motion of the mouse and of all the drivers of a display, considers the parameters of *stationary* oscillations.

Since users are also able to modulate such parameters in real time, there is room for designing novel techniques based on the *dynamic* exploitation of sustained oscillations. One example is Rub-Pointing.Click [21], which relies on diagonal to-and-fro movements of the finger. This technique allows a zooming progression to be controlled in closed loop by a modulation of the oscillation frequency, each reversal of the rubbing gesture resulting in a doubling of viewing scale. The amplitude of the gesture, which must be more than 3 pixels and less than 50 pixels, serves to decide whether the movement actually corresponds to a rubbing gesture or a traditional drag.

THE CYCLO* APPROACH

Our approach focuses on elliptic oscillatory gestures as in [11], in an attempt to generalize previous techniques based on straight or circular periodic gestures. A circle and a line segment are limiting cases of an ellipse, with an eccentricity of 0 and 1, respectively. Elliptic oscillatory gestures have several interesting properties. First, they make it theoretically possible to control up to seven mathematically independent variables. An ellipse has five geometrical DoF in the plane, which may be described as: (1) major axis orientation relative to the *x* axis, (2) amplitude along its major axis, (3) eccentricity (the minor/major axis amplitude ratio), (4) and (5) *xy* ellipse location. Provided the approach is kinematic, we have another two DoF, (6) frequency and (7) drawing direction (clockwise or counterclockwise). Indeed, it would be unreasonable to expect users to be able to vary all these variables orthogonally if only because a number of preferred couplings are known to exist (e.g., between amplitude and frequency in walking [9]). Nevertheless, elliptic control offers an abundant set of DoF for designing novel techniques for interaction on sensitive surfaces.

It is remarkably easy for humans to oscillate their hand in a quasi-harmonic fashion [12]. Once such an oscillation is settled, its parameters can be modulated in real time. The important facts seem to be the following:

- Oscillatory motion allows continuous, closed-loop (rather than intermittent, ballistic) control over the variables to be set or manipulated.
- Periodical motion is spatially unbounded, making it possible to avoid clutching.
- Fairly small oscillations, which save real estate, can work as input gestures. This is because most dynamic properties of an oscillation are scale-independent and an oscillation is an essentially stationary phenomenon that does not tend to expand with the elapse of time. This property is particularly useful for small surfaces.
- The geometric and kinematic properties of an oscillation being location independent, an oscillatory gesture can be performed opportunistically in any convenient area of the sensitive surface. Alternatively, however, interaction techniques may be designed that consider the *xy* location of the ellipse center.

CYCLOPAN

CycloPan was designed for panning large documents. Like Rub-Pointing.Click [21] it relies on to-and-fro oscillatory gestures but exploits more DoF (Fig. 2). Rubbing only considers two orientations of the gesture (NW-SE and SW-NE diagonals). CycloPan considers stroke orientation ϕ, stroke amplitude A, and oscillating frequency ($F = 1/(2*(T_2 - T_1))$, where T stands for time), these variables playing the roles of a bicycle's handlebar (orientation control), pedals (distance control) and gears (gain control), respectively.

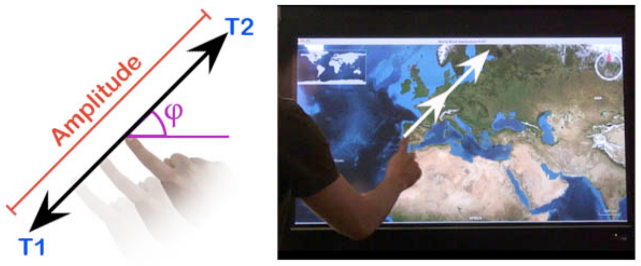

Figure 2: CycloPan (see explanations in text).

Finger motion during the first stroke of a CycloPan oscillation has the same effect as a standard drag gesture, with direct motor-visual space correspondence and a gain of 1. The change occurs at the first direction reversal. The view continues to be panned in the initial direction although the gesture is now opposite (Fig. 2). Direction correspondence is restored for one stroke after the second reversal, then the direction is again reversed, and so on. This mechanism mimics a crank, which converts circular motion into linear motion except that here the 'rotation' is along a flattened circle (eccentricity of 1). To avoid triggering CycloPan involuntarily, this mechanism is activated only if the mean speed of the first stroke exceeds 50 pixels/sec. Conversely, it is turned off when the finger stops moving for a 300ms delay (or, of course, if it is lifted).

A nice feature of this design is that it allows continuous speed control along with steering. No movement in motor space is wasted, as both back and forth strokes contribute to the pan. This contrasts with the standard drag, which requires clutching when the finger approaches an edge of the sensitive surface. The clutching problem being solved, motion in virtual space may cover as large distances as one desires regardless of the size of the input device and despite the amplitude limitations suffered by a user's hand.

Another important property is that CycloPan is compatible with standard drag and flicking. A drag is just a degenerate case of a CycloPan gesture where no reversal occurs. A flick gesture can be detected in the usual way, through kinematic analysis at release. Users can thus freely choose which of the drag, the flick, and the CycloPan gestures is the most appropriate technique given their task.

In CycloPan the panning gain is controlled by varying the frequency of the oscillatory movement. Initially set to 1, the gain is updated at every reversal and computed as a running average over the last three half-cycles so as to avoid jerky speed variations:

$$g_i = \max(1,\ k * \frac{1}{3} * \sum_{j=i-2}^{i} f_j)$$

with g_i being the gain of the i^{th} stroke, f_j the frequency for the j^{th} stroke (i.e., the inverse of the time needed to draw it) and k a constant, whose value was optimized empirically and set to 2/3. Hence, n being the number of strokes of a

CycloPan gesture, the total distance d_n in the visual space is the sum of the amplitudes a_i in the motor space modulated by the corresponding gains:

$$d_n = \sum_{i=1}^{n} g_i * a_i$$

The frequency for controlling the gain was optimized empirically. The dynamically adjustable gain makes it possible to go faster when long distances must be covered. Combined with the absence of clutching, this property makes CycloPan well adapted for long displacements without requiring specific hardware (e.g. RubberEdge [8] an efficient solution that combines position and rate control but at the cost of extra equipment). Gain control is comparable to the so-called mouse pointer 'acceleration'. This feature is by definition not available on absolute input devices such as touch-screens because this would break the one-to-one input/output correspondence, except if the absolute device is used as a relative device (touchpad). As CycloPan does not involve one-to-one input/output correspondence, except for the first stroke, it extends the advantages of pointer acceleration to interactive surfaces.

EXPERIMENT 1: EVALUATION OF CYCLOPAN

The aim of Exp. 1 was to compare CycloPan with two well-known panning techniques, the basic drag and the flick. We used a realistic task that involved a mixture of pointing and steering. The participants had to follow a continuous labyrinth linking consecutive nodes up to a final target. This task mimics what users have to do when searching for some location on a road or metro map. While we expected CycloPan to outperform the simple drag (our bottom baseline), we were curious about how well our novel technique would compare with the popular flick (our top baseline).

Although we designed Cyclo* techniques primarily for touch screens, we found it useful to check whether they could also improve panning on a laptop touchpad, one of the most popular input device nowadays.

Method

Equipment. The experiment was run on a 2.4 GHz Intel Core 2 MacBook Pro Notebook. The experimental software was implemented with Java Swing, and refreshed at 60Hz. The display used full screen mode on the native 17' screen of the laptop with a 1,920 x 1,200px resolution. Observation distance was 60cm, the advised distance for ergonomic usage [7]. The users were invited to use the native touchpad of the laptop, 10cm wide and 4.8cm high.

Display and Task. The participants' task was to pan the document so as to find and validate a sequence of 16 small circular targets placed at the summits of a broken line (Fig. 3). The path linking the 16 targets formed a labyrinth whose shape was variable across trials but whose total length was a constant 46,240px, and so a small proportion of the labyrinth was visible in the view.

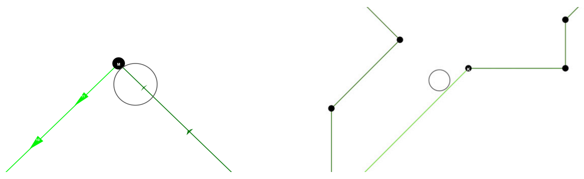
Figure 3: Two views of the labyrinths that users had to follow.

The line segment leading to the next target appeared in green, with superimposed arrows whose length was proportional to current target distance, so that the participant were informed of the distance that remained to be covered before the target entered the view. The direction of any target with regard to the previous one was chosen randomly with the following constraints: a) all combinations of eight directions (45°, 90°… 360°) by two distances (SL or 4 x SL, SL being the screen length) were represented exactly once; b) the turn angle at a summit was neither 0° nor 180°. One consequence of these constraints was that all labyrinths ended up at their start point, forming a closed circuit.

A target was validated if it was crossed by a dedicated aiming circle 275px in diameter permanently displayed at screen center. To make sure participants would not rush across targets without paying them a visit, we asked them to read aloud the small letter, randomly chosen by the program, that appeared in the middle of each target. The answer was checked by the experimenter and the participant had to return to the missed target in case of an error. Hence, the net error percentage was 0%, allowing task completion time to be held as a reliable global measure of performance.

Techniques. Three techniques were investigated, the basic drag, the flick and CycloPan. The flick technique was set to reproduce its implementation on the iPhone.

Practice and Procedure. Participants were offered up to five warm-up blocks per technique. They were allowed to take less practice if they felt they were ready for the experiment.

The experiment was organized in blocks in each of which the participant had to visit 16 targets. Each technique was used in three consecutive blocks with different labyrinths, Latin squares serving to balance the labyrinth sequences.

Twelve adult volunteers (9 male, 3 female, all right-handed) participated in a single session, which lasted about half an hour. We used a fully within-participant design which overall involved 16 targets x 3 techniques x 3 blocks = 288 individual target acquisitions per participant.

Results and Discussion

The mean performance measures are shown in Fig. 4. An ANOVA with technique and trial-block number as factors revealed statistically significant effects of technique ($F_{2,22}$=14.28, *p*<.003) and practice ($F_{2,22}$=6.75, *p*<.03), with no significant interaction ($F_{4,44}$=1.44, *p*=.25). On average over the three blocks, performance was fastest with CycloPan (53.2s) and slowest with the simple drag (69.3s). Post-hoc tests showed that if both CycloPan and flicking significantly surpassed the drag, the difference between the two elaborate techniques (53.2s vs. 57.9s) was not statistically reliable.

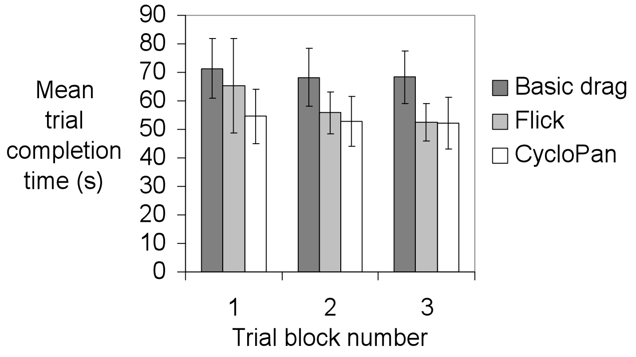

Figure 4: Time performance for each technique and each block of the experimental session.

Subjective Preferences. The participants were requested to rank the three techniques by order of preference at the end of the session (Table 1). The drag technique (mean rank = 2.75) was judged worst by a vast majority of participants. The other two more sophisticated techniques were about equally popular: for flicking and CycloPan the mean ranks were 1.67 and 1.58, respectively, the difference being far from statistical significance.

	1st	2nd	3rd
Drag	1	1	10
CyclePan	6	5	1
Flicking	5	6	1

Table 1: Number of times each technique was ranked first, second, and third.

The outcome of this experiment is consistent with our hypothesis that CycloPan is more efficient and more pleasing than the basic drag. One plausible explanation is that the latter technique suffers from clutching, even though the touchpad of our experiment happened to be pretty large (10 x 4.8cm).

The statistical evidence does not allow us to claim that in this experiment CycloPan objectively outperformed flicking, nor that it was subjectively preferred: the two techniques have similar merits. CycloPan was easy to learn, scoring best at the outset of the session (Fig. 4). This is surprising as CycloPan was quite novel unlike the flick, more or less familiar to all our participants.

At any rate, whether or not CycloPan outperforms flicking is a somewhat secondary concern. In our view CycloPan is not meant to replace well-known and well-tried techniques such as flicking or the basic drag, but rather to enrich the users' resources for input expression. As already emphasized, CycloPan, flicking and the classic drag are fully compatible with one another, so that the user may be allowed to freely choose the most appropriate technique for the task at hand. CycloPan can in fact be viewed as a method of *augmenting* the classic drag.

A reasonable guess is that the basic drag and flicking should be best suited for small and large displacements, respectively, while CycloPan should be especially useful for displacements with an intermediate length and for browsing documents. Like basic drag, but unlike flicking, CycloPan offer continuous control over panning speed. Contrary to the two other techniques, CycloPan motion goes on uninterruptedly, the user being able at every single instant to slow down or accelerate the panning motion.

CYCLOZOOM+: INTEGRATED PAN AND ZOOM

View panning is needed when the user wishes to visualize some region of virtual space located not too far away from the view. If the region of interest is located at a distance several orders of magnitude away from the currently visible region (e.g., in another continent, using Google Earth), then zooming is in order [13].

Multi-scale navigation from one focus of interest to another typically involves a *zoom-out* and a *zoom-in* phase. First, the user must zoom-out to allow the region of interest to enter the view, the target element being usually too small at first for being displayed. Then zooming-in may start, and it will have to be continued up to the scale at which the target becomes visible. With most techniques zooming-in requires periodic panning adjustments because every zooming-in step causes the current distance between the target element and the focus of expansion to increases non-linearly [13, 14]. Unlike the initial zooming-out phase, zooming-in thus demands an intricate coordination of zooming and panning on the part of the user [5].

From the above two practical lessons may be learned. One is that we can use, with little risk, a reasonably higher *gain coefficient* for the control of zooming-out than zooming-in, provided that the user remains able to control the zoom level continuously. Since everything—in particular the target region—converges on the contraction focus during the zoom-out, zooming-out is a fairly easy task that can be carried out in a wholly ballistic way. This explains why in our implementation of CycloZoom+ the zooming speed for a given velocity of the finger is more than twice higher for zooming-out than zooming-in. The other lesson is that the extent to which the two components of zooming-in are integrated is an important aspect of multi-scale navigation techniques [5, 17]. In CycloZoom+ the zooming-in and the pan actions can be carried out *in parallel*. During the zooming-in approach the user can continuously adjust the location of the expansion focus. This feature of CycloZoom+ is a significant improvement over the Rub-Pointing.Click technique [21], which requires users to lift their finger from the screen whenever the expansion focus needs to be shifted (and such an event may occur several times in a long enough zooming-in phase).

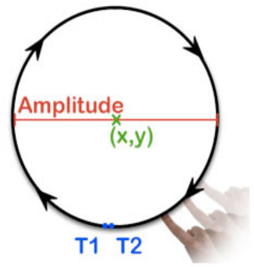

Figure 5: The DoF of CycloZoom+ (see explanations in text).

CycloZoom+, like CycloPan, is based on the model of an ellipse but it requires approximately circular gestures. In fact, by varying gesture eccentricity, the user can trigger either CycloPan if eccentricity falls in the [0.5, 1] interval (more oblong ellipses) or CycloZoom+ if eccentricity falls in the [0, 0.5] interval (rounder ellipses). Five DoF are then used to control interaction (Fig. 5):

- The *zooming direction* (in vs. out) depends on the direction of finger motion along its circular path: consistent with the metaphor of a screw, zooming-in results from clockwise circling and zooming-out from counterclockwise circling.
- The *zooming speed* depends on circling frequency: the faster the angular velocity of the finger, the faster the zoom progression (the ratio of zooming speed to the angular frequency of the circling motion being higher for zooming-out than zooming-in).
- The *zooming accuracy* depends on the circling radius. Users can modulate their angular speed by varying the radius of their circling gesture. When they feel they are zooming too fast or too slow, they may use the radius as an adjustable gain parameter for the control of zooming speed. This means capitalizing on the so-called vernier effect [10, 26, 27]: for a given tangential speed, the smaller the radius, the faster the angular velocity. CycloZoom+ thus offers a dynamically adjustable resolution that makes it possible either to control large scale variations very efficiently or to make very precise scale adjustments when needed (for instance, for adjusting the font size at the exact desired value at the end of a zooming operation).
- The *expansion* or *contraction focus* is located at the center of the circular gesture. This is particularly useful in the zooming-in phase where users simultaneously pan by shifting the location of the circling gesture. By continuously adjusting the position of the expansion focus so that the target region remains within the circle, they can correct in real time the positioning error revealed by the zoom-in, with no need to alternate between a zooming and a panning mode.

Implementation

The best-fitting ellipse computed online from the circling gesture is updated twice a period. The first computation occurs at the first reversal, i.e., at the point where the distance from the initial press point starts to decrease. The second reversal is calculated similarly, but by considering the distance from the first reversal point, and so on.

The initiation of a circling gesture is interpreted in CycloZoom+ (as well as CycloPan) as a simple drag. Thus a zooming action will always start with a small panning effect. If a flat ellipse is recognized, CycloPan is activated at the first reversal; if, however, a round ellipse is recognized, the program waits for the *second* reversal (i.e., for completion of the first cycle) to activate CycloZoom+. This makes it possible to specify (for the user) and to determine (for the program) the ellipse center, and hence the expansion focus, more accurately. This option also cancels the effect of the irrelevant first drag, because when a full circle has been drawn, the document has returned to its initial position.

The scale of the view is updated each time the user moves the mouse. It is computed as follows:

$$s_i = s_{i-1} * (1 + K * a_i)$$

with s_i being scale at step i, a_i the angular variation between steps i and $i-1$, and K a constant that was empirically set to 0.14 for zooming-in and -0.32 for zooming-out.

For each new sample from the input an estimate of angular displacement is obtained by dividing the distance between the points at steps i and i-1 by the perimeter of the circle (re-estimated at each half cycle, as already mentioned).

EXPERIMENT 2

This experiment aimed to compare the efficiency of CycloZoom+ and Rubbing for navigating on a map. We resorted to the multi-scale pointing paradigm of [13,14], whose major advantage is to be accurately defined and to deliver interpretable measures. We needed a tough enough challenge for our two techniques, and so we decided to use a target-reaching task with a fairly high index of difficulty *ID* = 15 bits.[1] For example on Google Earth such an *ID* would correspond to the task of reaching a 500m target (say, a stadium) on the opposite side of the planet.

Method

Equipment. The computer was the same as for Exp. 1 but we used a SmartBoard, a vertical, front-projection 121x90.5cm interactive whiteboard with a display resolution of 1024x768px.

Display and Task. The participants were to carry out a multi-scale pointing task [13,14] while standing in front of the vertical whiteboard, using a single finger. They had to reach and validate a series of 8 circular targets 60px in diameter each of which was separated from the preceding

[1] Initial distance D_0 was 2,000,000px and the target width W was a constant 60 px. According to the Shannon formula of Fitts' law, $ID_0 = \log_2(D_0/W+1) = 15.02$ bits.

by a distance of 2 million pixels. They were instructed to complete the sequence as fast as possible, using the zooming and panning facilities made available to them.

Once the participant had validated a target with a finger tap, this target disappeared, being replaced instantly by another target in one of the 8 randomly-chosen cardinal directions, at a distance of 2 million pixels. Of course the new target was not visible in the view at first. A pattern of concentric circles centered around the target was displayed so that the participant could know in which direction (s)he had to go [13]. Once, after an appropriate amount of zoom-out and pan the target area eventually entered the view, it was materialized by a white disc (the flag) whose size (60px, 7cm) was constant. These visual aids, the concentric circles and the flag, were provided so as to avoid the so-called desert-fog problem [16]. In the end of the zoom-in phase, the scale-independent white flag was replaced by the rescalable red target, meaning the actual selection could take place.

Techniques. We investigated CycloZoom+ against Rubbing (Rub-Pointing.Click [21]) whose zoom coefficient was set to 2 for both zooming-out (NW-SE diagonal) and zooming-in (NE-SW). The basic drag was always available in the experiment, and so when using CycloZoom+ participants had at their disposal two ways of obtaining a panning effect: they could simply drag (pure pan) or shift the expansion focus while zooming (combined pan).

Procedure. The experiment was organized in blocks in each of which the participant had to select 8 targets. Each participant had to perform 8 blocks per technique, in a balanced order. Half of the participants started with CycloZoom+ and the other half with Rubbing.

Twelve adult volunteers (8 male, 4 female, all right-handed) participated in a single 40-mn session. We used a fully within-participant design which overall involved 8 targets x 2 techniques x 8 blocks = 128 individual target acquisitions per participant.

Results and Discussion

Performance Measures. Panel D of Fig. 6 illustrates target-acquisition time measured from each successful target tap to the next. This measure corresponds to errorless performance since any final tap that failed to touch the target had to be immediately repeated. In Fig.6A, 6B, and 6C this measure is parsed into its three serial components, mean zooming-out time (up to max viewing altitude), transition time (the time spent at max altitude), and zooming-in time (the time spent at diving to the target down to the selection tap).

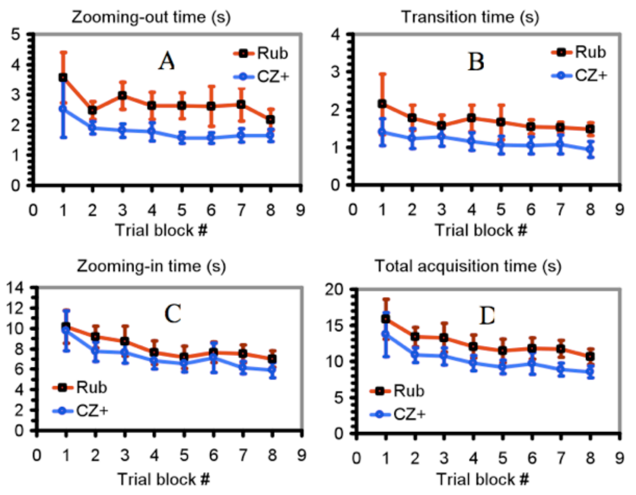

Figure 6. Time performance for each technique and each block of the session. Total acquisition time (panel D) is decomposed into three components, zooming-out time (A), transition time (B), and zooming-in time (C).

The CycloZoom+ technique surpassed Rubbing (10.17s vs. 12.53s on average over the eight trial blocks), allowing a 18.9% time saving ($F_{1,11}$=11.87, p=.005). Performance improved throughout the session ($F_{7,77}$=15.46, p=.002), with no significant block x technique interaction. The absence of asymptotes in Fig. 6D suggests that with more practice performance would have continued to improve for both techniques.

Unsurprisingly [14], zooming-in was the longest component of target acquisition time. The superiority of CycloZoom+ was less marked for zooming-in (a non-significant 11.3% time saving, $F_{1,11}$=2.72, p=.12) than for zooming-out (a 33.5% time saving, $F_{1,11}$=22.1, p<.001) and for the transition phase (a 32% time saving, $F_{1,11}$= 14.2, p=.003). The technique effect in the ANOVA fell short of significance for zooming-in but non-parametric tests were quite significant: the technique difference was 10/12 times in favor of CycloZoom+ and a two-tailed sign test delivers a fairly safe p=.019.

	$F_{1,11}$	p
Blocks 1 though 8	2.72	.128
Blocks 2 though 8	3.80	.077
Blocks 3 though 8	3.20	.101
Blocks 4 though 8	3.58	.085
Blocks 5 though 8	4.38	.060
Blocks 6 though 8	5.21	.043
Blocks 7 though 8	14.42	.003
Block 8	7.05	.022

Table 2. *F* statistic and probability of a error for the technique effect on Zooming-in time.

A closer look at individual performance data revealed that within- and between-participant variability dropped substantially during the session. As shown in Table 2, the *F* statistic increases in a nearly monotonic fashion as a larger and larger proportion of initial blocks are excluded from the ANOVA, suggesting that a certain amount of practice is needed to fully exploit the advantages of the CycloZoom+ technique. As said above, zooming-in requires periodic panning adjustments and the participants could simply drag (with both techniques) or shift the expansion focus while zooming (with CycloZoom+). The mean amplitude of these drag gestures were pretty small: 18.0cm CycloZoom+ and 21.8cm for Rubbing on a 121x90cm SmartBoard, which explains why CycloPan, which was available in the CycloZoom+ condition, was in fact never used.

Subjective Preferences. Eleven of our twelve participants reported they preferred CycloZoom+ over Rubbing, one reporting the opposite opinion (p<.005, two-tailed sign test). Fig. 7 illustrates the subjective evaluations of the two techniques averaged over the 12 participants, using a 1-5 scale where 5 means "I totally agree". CycloZoom+ was judged simpler to use, more pleasing, more accurate, and less tiring.

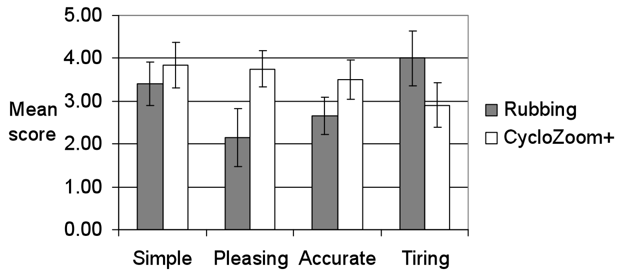

Figure 7. Subjective judgments.

In sum, CycloZoom+ allowed faster multi-scale navigation performance than Rubbing and was quite favorably judged by our participants. These findings may reflect several causes discussed below.

The first major difference between the two techniques is that Rubbing gestures zoom incrementally in discrete steps while CycloZoom+ allows users to zoom continuously as long as they move their finger. Moreover, thanks to the Vernier effect, users can adjust the zooming speed and precision by changing the radius of their circling gestures. These properties have two important consequences.

First, they make it possible to boost the zooming-out rate (which in our experiment was 2.3 times higher for zoom-out than zoom-in for CycloZoom+) because users have full continuous control over the zooming progression and zooming speed. This contrasts with the Rubbing technique, which varies the zoom level by discrete steps. Several participants reported that the zoom coefficient of 2 that was used in our experiment, as in [21], felt high. Hence boosting the zooming-out rate beyond this value would have been problematic for Rubbing gestures, unlike CycloZoom+. This larger zooming-out rate probably explains why CycloZoom+ outperformed Rubbing to an impressive extent during the zooming-out phase, with a 33.5% time saving.

The second consequence of continuous control over progression and speed is that it makes possible to finely tune the zoom level, for instance for displaying characters exactly at the desired size, and in the end of a zooming action. It is an interesting property of CycloZoom+ that a finer and finer variation of scale can be produced by tracing an expanding spiral (Fig. 8).

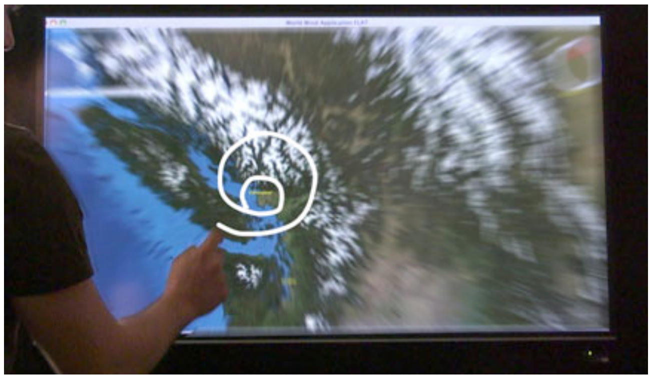

Figure 8. Expanding spiral for finer variation of scale.

Another key property of the CycloZoom+ technique is that it allows participants to pan and zoom simultaneously during the zoom-in phase by changing the focus of expansion continuously, whereas Rubbing requires several lifts to modify its position. This property also applies for the transition phase (between zooming-out and zooming-in) where one can reverse the zooming direction, while panning, without lifting the finger. Also, as remarked in [21], circling around a target area seems quite natural, all the more so if the circular gesture is made to enclose the expansion focus. These points probably explain the efficiency of CycloZoom+ for the transition and zooming-in phases, although some practice seems necessary to fully master the technique in the latter case.

A possible weakness of circular gestures relative to linear strokes is that they require the user to draw a full circle before the system switches to the appropriate mode. However, our experience is that users generally draw fairly small circles, and neither the delay nor the visual perturbation seems to be problematic.

Finally, an important detail is that our implementation of the Rub-Pointing.Click technique was symmetrical, in the sense that the "rubbing-in" and "rubbing-out" gestures served to zoom-in and out, respectively. This is unlike [21], who explored an asymmetrical version where rubbing-out served to reset the zoom, meaning a return to the global view in an all-or-none fashion. We made this choice to provide participants with full control over the scale level, on the grounds that in real situations the global view is context dependent. If one is interested in a specific region of a map, obviously a switch straight to the global view of the planet would be irrelevant and perturbing since the user would have to zoom-in back to the regional level of scale. Zooming-in being a costly operation, zoom-out overshoots seem undesirable. Besides, an additional gesture could serve to reset the zoom. Such a gesture would of course be compatible with the CycloZoom+ gestures as will be shown in the following section.

EXTENSIONS AND APPLICATIONS

CycloPan and CycloZoom+ are but two specific instantiations of Cyclo*, a fairly general principle for the design of interaction techniques. The parameters of an oscillatory gesture might serve to control any continuous variable like a position, a velocity, a color, a sound level, etc. Since an oscillation might be meant to control, say, the sound volume or screen brightness, one problem is to let the system know what particular variable the cyclical gesture is supposed to control. That may be solved by defining a taxonomy of ellipses whose classes will be associated with different modes: a simplified instance is given in Fig. 9.

So far we have focused on two limiting cases of elliptic motion, circular vs. linear to-and-fro gestures, but finer classifications of ellipses may be thought of using, among others, the eccentricity criterion (Fig. 9, left). Taking ellipse orientation into account, the tilt parameter can be controlled.

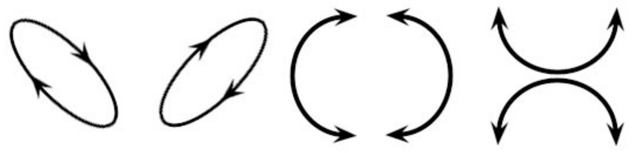

Figure 9: Some possible contrasts in a taxonomy of oscillatory gestures. Left: 0.5 eccentricity ellipses; right: half-ellipses.

One might also use half elliptical shapes (Fig. 9, right). Such gestures cannot presumably be performed at as high frequencies as oscillatory strokes or circles, but they might serve for occasional commands (e.g., resetting the zoom in CycloPan). More work is needed to determine how many of these gestures could be distinguished and if they should follow cardinal vs. diagonal directions (which may depend on the surface type, as the constraints probably differ between a whiteboard and a touchpad).

Techniques of the Cyclo* class could help for other tasks than document navigation, for instance for extending a selection beyond the limits of an application window. When one wants to extend one's selection, zero-order control is typically replaced by rate (first-order) control as soon as the cursor reaches the window edge. But rate control is not very adequate for determining where the selection should stop. CycloPan might be a solution. The selection being triggered as usual, a 'CycloSelect' mode might be chosen by oscillating the cursor and probably tracing some specific kind of periodic gesture. As a sustained oscillation allows closed-loop control over the progression, this would result in a fine control over the size of the selection zone, the document being automatically scrolled if needed.

Another possible application would consist of using oscillations for navigating among applications (as a substitute for the "Alt-Tab" shortcut) or among overlapping windows. Using reversals to cycle in a list might be especially useful on very small and very large surfaces where users miss a keyboard, modifying keys and hotkeys. Gestures like those of Fig. 9 could serve that purpose. Additionally, this could also allow to perform drag and drop among overlapped windows, the circular movement being then used to switch from one window to another in the window stack. The beginning of the gesture would start on the object to drag in the first window and end on the drop location in the second window.

CONCLUSION AND PERSPECTIVES

We have presented CycloPan and CycloZoom+, two navigation techniques which exploit several degrees of freedom of elliptic motion. A few techniques resorting to oscillatory motion have already been described in the literature, but the proposed approach attempts to go further by exploiting elliptic DoF for controlling continuous variables. Improving over previous scrolling techniques based on oscillatory movements (e.g., [20, 26, 27]), CycloPan makes it possible to pan in 2D rather than 1D space with a closed-loop control over the gain. Similarly, CycloZoom+ expands zooming techniques such as [21] with more DoF being exploited. CycloZoom+ makes it possible to zoom and pan simultaneously, while controlling the scale with a dynamically adjustable level of resolution. We emphasize the fact that the techniques we describe do not compete with others: other parameters (e.g., rotations, tilts) can be controlled by means of other gestures.

Unlike previous authors, we associate oscillatory strokes with panning and oscillatory circles with zooming. Pilot experiments have shown that this mapping looks most natural to users: as strokes indicate a direction, they are particularly well suited for 2D panning; as circles indicate a center, they are particularly well suited for marking the focus of expansion/contraction of a zooming effect.

While this paper mainly focused on navigation tasks, the proposed approach, which we named Cyclo*, is quite general and can be used for many other tasks that involve the setting of continuous (as well as, of course, discrete) variables. Elliptical movements make it theoretically possible to control up to seven variables. Even though users are unlikely to be able to vary any two of them orthogonally, elliptic control provides a precious reservoir of DoF for implementing rich interaction techniques that go beyond the mere field of navigation techniques. We believe these techniques are of interest in a broad variety of hardware contexts (e.g., touchpad input on laptops, direct touch-screen interaction on handhelds and enlarged displays). More research is needed to list among the many DoF of hand oscillations the particular subsets that enjoy the most independence from one another and hence qualify as promising input controllers for HCI.

ACKNOWLEDGMENTS
We gratefully acknowledge the financial support of the Cap Digital ENEIDE project which was funded by Région Ile-de-France and DGE. We thank the anonymous reviewers of this article for their relevant critiques and recommendations. Many thanks to G. Bailly, A. Tabard and A. Roudaut, who helped us in multiple ways, and to all our participants.

REFERENCES
1. Aliakseyeu, D., Irani, P., Lucero, A., and Subramanian, S. (2008). Multi-flick: an evaluation of flick-based scrolling techniques for pen interfaces. Proc. CHI '08. ACM Press, 1689-98.
2. Arthur K.W., Matic N., Ausbeck P. (2008).Evaluating Touch Gestures for Scrolling on Notebook Computers. CHI'08 Extended Abstracts, 2943-2942.
3. Bailly, G., Lecolinet, E., and Nigay, L. (2008). Flower menus: a new type of marking menu with large menu breadth, within groups and efficient expert mode memorization. Proc. AVI '08. ACM Press, 15-22.
4. Benko, H., Wilson, A., and Baudisch, P. (2006). Precise selection techniques for multi-touch screens. Proc. CHI'06, ACM Press, 1263-1272.
5. Bourgeois, F. & Guiard, Y. (2002). Multi-scale pointing: Facilitating pan-zoom coordination. Ext. abstracts CHI'02, 758-59.
6. Buxton, W. (1990). A three-state model of graphical input. Proc. INTERACT '90. Amsterdam: Elsevier Science Publishers B.V. (North-Holland), 449-456.
7. Cardosi, K., Murphy, E. (1995). Human factors in the design and evaluation of air traffic control systems. Technical report, Federal Aviation Administration, 1995
8. Casiez G., Vogel D., Pan Q., Chaillou C. (2007) RubberEdge: Reducing Clutching by Combining Position and Rate Control with Elastic Feedback Proc UIST'07, ACM Press, 29-138.
9. Danion F., Varraine, E, Bonnard, M, Pailhous, J. (2003). Stride variability in human gait: the effect of stride frequency and stride length. *Gait and Posture 18*, 69-77.
10. Evans K.B., Tanner P.P., Wein M. (1981). Tablet-based valuators that provide one, two, or three degrees of freedom, ACM SIGGRAPH Computer Graphics, v.15 n.3, 91-97.
11. Fekete, J.D., Elmqvist, N., & Guiard, Y. (2009). Motion Pointing: Target Selection using Elliptical Motions. Proc CHI'09, ACM Press.
12. Guiard, Y. (1993). On Fitts' and Hooke's laws : simple harmonic movement in upper-limb cyclical aiming. Acta Psychologica, 82, 139-159.
13. Guiard, Y., Bourgeois, F., Mottet, D., Beaudouin-Lafon, M. (2001). Beyond the 10-bit barrier : Fitts' law in multiscale electronic worlds. Proc. IHM-HCI 2001, Springer, 573-587.
14. Guiard, Y., Beaudouin-Lafon, M. (2004). Target Acquisition in Multi-Scale Electronic Worlds. IJHCS, 61, 875-905.
15. Guimbretière, F., Winograd, T. (2000). FlowMenu: combining command, text, and data entry. Proc. UIST'00, ACM Press, 213-216.
16. Jul, S., Furnas, G. W., 1998. Critical zones in desert fog: Aids to multiscale navigation. Proc. UIST'98. ACM Press, New York, pp. 97-106.
17. Jul, S. (2003). This is a lot easier! Constrained movement speeds navigation. Ext. Abstracts ACM CHI'03. ACM Press, New York, pp. 776-777.
18. P. N. Kugler and M. T. Turvey (1987). Information, natural law, and the self-assembly of rhythmic movement. Lawrence Erlbaum Associates..
19. Kurtenbach, G. and Buxton, W. (1991). Issues in combining marking and direct manipulation techniques. Proc. UIST'91, ACM Press, 137-144.
20. Moscovich, T., Hughes, J.F. (2004). Navigating Documents with the Virtual Scroll Ring. Proc UIST'04, ACM Press.
21. Olwal, A., Feiner, S., Heyman, S. (2008). Rubbing and Tapping for precise and rapid selection on touch-screen displays. Proc. CHI'08, ACM Press, 295-304.
22. Oniszczak, A. and MacKenzie, I. S. (2004). A comparison of two input methods for keypads on mobile devices. Proc. NordiCHI '04, ACM Press, 101-104.
23. Parhi, P., Karlson, A., Bederson, B. (2006). Target Size Study for One-Handed Thumb Use on Small Touchscreen Devices. Proc. MobileHCI'06. 203-210.
24. Pook, S., Lecolinet, E., Vaysseix, G., Barillot, E. (2000). Control menus: execution and control in a single interactor. Proc. CHI'00, ACM Press, 263-264.
25. Roudaut, A., Lecolinet, E. Guiard, Y (2009). MicroRolls: Expanding Touch-Screen Input Vocabulary by Distinguishing Rolls vs. Slides of the Thumb. Proc CHI'09, ACM Press.
26. Smith, G.M., schraefel, m.c. (2004). The radial scroll tool: scrolling support for stylus- or touch-based docu-ment navigation. Proc. UIST'04. ACM Press, 53-56.
27. Smith, G., schraefel, m. c., Baudisch, P. (2005). Curve dial: eyes-free parameter entry for GUIs. Proc. CHI'05 Extended Abstracts, ACM Press, 1146-1147.
28. Williamson, J., Murray-Smith, R. (2004). Pointing without a pointer. Proc. CHI'04, ACM Press, 1407–10.
29. Zeleznik, R., Forsberg, A. (1999). UniCam - 2D gestural camera controls for 3D environments. Proc. I3D'99, ACM Press, 169-173.