Facet: A Multi-Segment Wrist Worn System

Kent Lyons, David H. Nguyen, Daniel Ashbrook, Sean White
Nokia Research Center
200 S. Mathilda Ave. Sunnyvale, CA 94086 USA
{kent.lyons, david.h.nguyen, daniel.ashbrook, sean.white}@nokia.com

ABSTRACT
We present Facet, a multi-display wrist worn system consisting of multiple independent touch-sensitive segments joined into a bracelet. Facet automatically determines the pose of the system as a whole and of each segment individually. It further supports multi-segment touch, yielding a rich set of touch input techniques. Our work builds on these two primitives to allow the user to control how applications use segments alone and in coordination. Applications can expand to use more segments, collapses to encompass fewer, and be swapped with other segments. We also explore how Facet concepts could apply to other devices in this design space.

ACM Classification Keywords
H.5.2 [Information interfaces and presentation]: User Interfaces. - Graphical user interfaces.

Author Keywords
wearable; multi-display; multi-segment touch; watch; bracelet

INTRODUCTION
Multi-screen displays are becoming increasingly common: from multi-screen desktop PCs or laptops with external displays to walls with dozens of screens. At the same time, the lowering costs of display technology is enabling an explosion of multi-display devices. Large screens are not the only ones increasing in number; small screens are multiplying as well. For example, Siftables [20] consist of multiple 1.5" display devices. When considering very small displays, one of the smallest practical screen sizes is the watch. Researchers have explored interaction with watches [2, 3, 4, 6, 16, 22, 24] and the challenges of input with other small screens [1, 25, 27]. While both watch interactions and numerous small devices have been investigated independently, multi-display watch-sized wearable systems have yet to be considered. In this paper, we introduce Facet, a segmented multi-display bracelet system for input and output.

Facet is segmented, comprised of multiple small I/O elements, and worn about the wrist (Figures 1 and 3). Facet's segmented nature enables affordances that would not be possible with a unified display. Users can create a one-to-one mapping of applications, activities, or people to specific segments; for example, one segment could be allocated to messaging and another to social network streams. This design creates an opportunity to make information that is important to the user quickly accessible. Much like the functional benefit for wearing a wrist watch to get time at a glance, Facet allows the user to glance at different segments that have information important to the user. Although segmented watches exist (*e.g.*, the Diesel DZ9024 has several watch faces), Facet is more analogous to an interactive charm bracelet where each segment is a physical artifact and has meaning for the wearer. Users can interact with and customize the system in a direct, tangible manner: gaps between segments allow quick display location by sight or touch, while applications can be easily swapped by replacing one segment with another. Finally, users can expand applications to encompass more segments when needed, and Facet's circular nature allows the user to view more content than there is physical space.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
UIST'12, October 7–10, 2012, Cambridge, Massachusetts, USA.
Copyright 2012 ACM 978-1-4503-1580-7/12/10...$15.00.

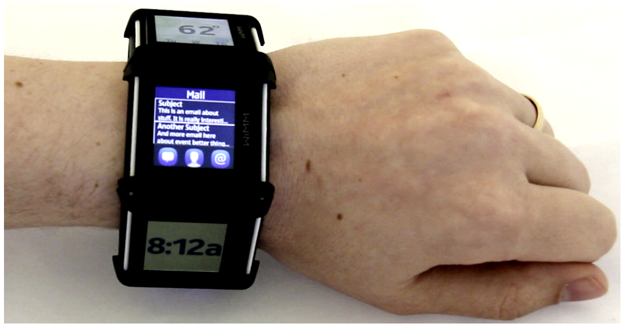

Figure 1: The Facet prototype, consisting of six WIMM One devices mounted in a bracelet such that they are removable.

Scenario
The following scenario illustrates the use of Facet. Karen uses Facet in both her work and personal life. Every morning, she slips on Facet and adds her work email and calendar segments (Figure 2a). Walking to work, she feels a buzz, and a glance shows an important email. Unable to see the entire message, Karen performs a open-pinch gesture to expand the email to the next segment (2b), where she sees she is needed in an afternoon meeting. Her calendar segment only shows the morning; expanding it to show the afternoon (2c), she finds she can attend. She pinches to collapse her email (2d), rotates Facet to find Messaging and sends a confirmation.

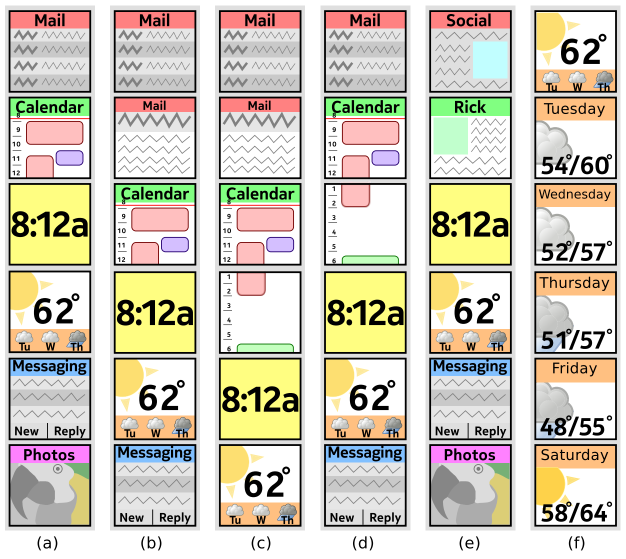

Figure 2: Illustration of our scenario. Each vertical strip indicates the state of Facet at a given point.

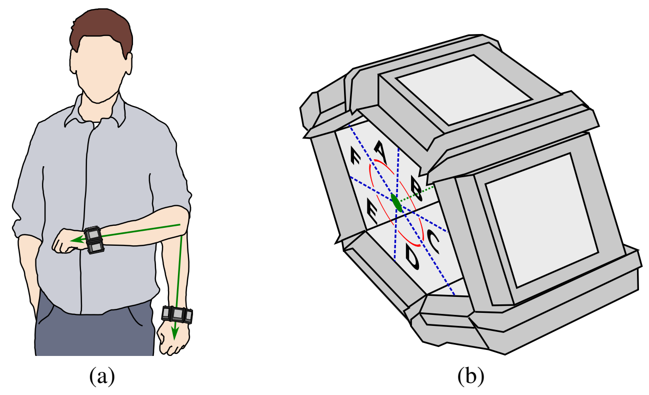

Figure 3: Image (a) shows the common axis shared by the segments as a green arrow, at two different arm angles. Image (b) shows how the relative order of the segments is determined. Each angle A–F (red solid curved lines) is 60° different from the preceding angle.

That evening, getting ready for dinner with her fiancé, Rick, Karen swaps her work email and calendar segments for a social networking segment and a *Rick* segment, which shows private information such as his location (Figure 2e). Walking to meet Rick, she thinks ahead to their weekend plans and checks the weather. Facet is only showing the next few days; tapping the weather and two other segments expands the application to the entire bracelet (2f). As she turns Facet, she sees the ten days forecast, with Facet cleverly allowing more days than segments. Satisfied, a quick shake of the wrist resets Facet to its default state and she continues to dinner.

Contributions

Facet makes a number of contributions to the field. Although multi-display systems are not new, there have been few investigations into multiple wearable displays. We introduce multi-segment touch, illustrate how the device can use each segment's pose to configure the system, and present a variety of interaction techniques. We discuss the design and implementation of the window manager-like Facet Manager (which allows the user to manipulate applications) and discuss how Facet's core principles could be extended to other systems.

RELATED WORK

Facet has commonalities with other wrist-based computing platforms. There have been numerous research projects investigating smart watches: researchers from IBM created a Linux-based watch [22]; Hutterer *et al.* created the infrastructure for a thin client watch [15]; and Martin examined the history of the watch including the transition from pocket to wrist-watches [19]. Smart watches have been commercially available for years (Hutterer *et al.* give an overview in [15]), and currently a new generation is entering the market such as WIMM Labs' "WIMM One" developer platform[1].

[1]`http://www.wimm.com`

One piece of previous research that has explored multiple watch-sized displays is Merrill *et al.*'s Siftables [20]. Individual Siftable elements have a form factor similar to Facet's segments; however, these were not intended to be worn or attached and used together. Furthermore the hardware itself and the interaction space are quite different. Siftables lack a touchscreen, and interactions between multiple elements are enabled through motion gestures (such as tilting or shaking) and IrDA-based neighbor sensing.

Other work has begun to explore multifaceted systems. Tuister has six displays arranged in a similar configuration as Facet [5]. Instead of being worn on the wrist, the Tuister displays encircle a handle. With Facet we explore touch screen interactions on small high resolution displays whereas Tuister explores a complementary interaction space using rotation. Poupyrev *et al.* also explore multifaceted interactions with their 20 sided D20 and consider how several faces can be seen and used simultaneously [21]. Their prototype utilized a physical 3D model that was tracked and 3D computer graphics simulated the display of D20.

Work on multi-screen systems is also relevant to Facet. There are different systems that allow users to bring together mobile devices such as phones [17] or tablets [11, 18]. Hinckley *et al.*'s Codex [12] uses two small tablets that work together while attached with a flexible interconnect, but also supports a configuration where each device can be removed. Some previous work has investigated multiple display elements on the wrist by using LEDs. Hansson and Ljungstrand's Reminder Bracelet used three LEDs for subtle notifications [8], while Williams *et al.* explored a six-LED bracelet where each of five of the lights represented the status of a different individual [26]. Similarly, Facet can dedicate a single segment to one type of content such as an individual's status.

Other systems have used inertial sensing for input, both deliberate and contextual: the work by Harrison *et al.* [9] is one of the earliest examples of intentionally moving a device for input. Rekimoto [23] utilized tilt for input for small screen devices and Harrison and Hudson [10] explored input using magnetic fields for wrist worn I/O . Hinckley *et al.* demonstrated early contextual response to sensor input such as switching between portrait and landscape modes [13]. Facet continues these traditions by enabling pose-dependent meaning for explicit input, as well as implicit automatic self-configuration with orientation.

Facet is made of components that are physically similar to watches and Siftables, and shares some of the multi-screen aspects of Codex and the segmented nature of Tuister. Facet explores the intersection of this prior work through multiple small interconnected wearable screens, leading to unique affordances different than a single smart watch, larger multi-screen systems, or the multi-device and multi-display non-wearable nature of Siftables and Tuister respectively.

FACET PLATFORM

Facet consists of an application stack with several components (Figure 4) and a hardware platform consisting of multiple WIMM One (hereafter simply "WIMM") devices. Each WIMM is a 667 MHz Android OS-based computer with a 1 x 1" (25.4 x 25.4 mm) 160 x 160 pixel capacitive touch bi-modal color display, an accelerometer and magenetometer, and WiFi and Bluetooth wireless communication. Each WIMM measures roughly 1.3 x 1.4 x .5" (32 x 36 x 13 mm) and weighs .78 oz (22g). The WIMMs form individual segments of Facet, which are joined together to form a bracelet (Figure 1). This design allows for quick removal, insertion, and placement of the bracelet segments. Our prototype allows up to six segments to be snapped in, but can easily adapted for wrist size or preference with more or fewer segments.

Coordination of multiple segments is handled by a distributed architecture. The segments use a custom inter-device communication framework that can use either WiFi or Bluetooth radios. This framework is built on top of the Open Sound Control protocol and provides a mechanism to pass human readable messages amongst our segments and a computer. Each segment provides input and output resources that are exposed remotely via a simple API. Currently each segment supports streaming sensor data (accelerometer, magnetometer, and touch). The software supports a very simple drawing interface where images can be shown, moved, removed, and animated. The software is implemented in Java as an Android application and the same software runs on all of the WIMMs.

We use a laptop running a custom Python program attached to the communication framework. This software provides a centralized point for processing sensor data and touch events. Currently, it also simulates application logic. Applications are implemented using sets of static images and by issuing the correct set of drawing commands via the framework to segments. This choice of implementation was to simplify development and this functionality could be distributed across the segments or placed into a master segment in the future.

INTERACTION COMPONENTS

Facet's construction enables two unique affordances that can be used as building blocks for interaction: pose detection and multi-segment touch.

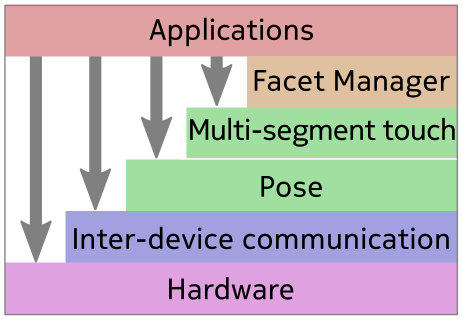

Figure 4: Facet system stack. Each layer takes advantage of the lower layers. Applications can use each layer directly. In this paper, we focus on the interaction components (pose and multi-segment touch) and the Facet Manager.

Pose

Using each segment's accelerometer and magenetometer, we extract orientation (pitch, roll and yaw) for each device with respect to a common coordinate system. This data allows us to dynamically calculate the relative ordering of segments in the bracelet, the angle of the user's forearm, and which segments are part of the bracelet.

Because the bracelet form factor imposes a circular arrangement, the segments all share a common axis running roughly along the user's forearm (Figure 3a). This property allows Facet to deduce the ordering of the segments by sorting each device's angle relative to their common axis. In our current six-segment prototype, each segment is rotated on this axis approximately 60° relative to the previous one (Figure 3b). The same data also provides information about the rotation of the bracelet around the wrist: by comparing the segments' common axis to the gravity vector extracted from the accelerometers, facet can calculate the orientation of the user's forearm. For example, when this common axis is approximately 90° from the gravity vector, the forearm is horizontal.

The pose data also provides an indication of membership—that is, whether an individual segment is physically attached to the bracelet. We compare each segment's orientation relative to the median orientation for all segments; segments within 1.2 medians are considered to be part of the bracelet. This technique can lead to false positives if disconnected segments are by chance oriented similar to the bracelet. Additional methods for finding connected devices could be applied, such as detecting common accelerations [14].

The order, arm angle, and membership data can be used in isolation by other parts of the system. They can also be used to form higher-level interaction primitives—for example, a quick rotation of the wrist back and forth is easy to detect as an oscillation in orientation about the forearm axis.

Multi-Segment Touch

Due to the limitations of the WIMMs, Facet segments do not individually support multi-touch. However, having several touch screens in direct proximity enables multi-segment touch, where each finger touches a different segment. Like traditional multi-touch we sense the number of touches and

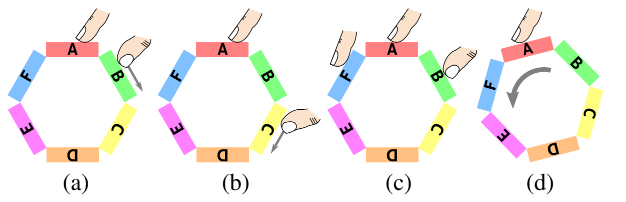

Figure 5: Side views of Facet showing touches: (a) is a two-finger pinch on adjacent segments; (b) is a two-finger pinch on non-adjacent segments; (c) is a three-finger chord; and (d) illustrates rotating Facet while touching a segment.

the position of the touch within each screen. However, the information generated by the Pose component provides additional information. First, because we know the relative ordering of the segments, we know relative touch position; for example, two adjacent segments, or two segments separated by one or two other segments. We can further augment the touch data with information about the pose of the entire bracelet. When Facet is being held in front of the user, the segments take on meaningful positions such as top, bottom and inside, and thus absolute segment positions can become meaningful. Using this information, a multi-segment touch on the top and bottom segments is distinct from the (relatively speaking) same touch on inside and outside segments (Figure 5a and b).

Multiple touches without movement form a chord; when moved, they become gestures. With three fingers (likely the maximum practical) the possibilities increase (Figure 5c). The user can also tap on one or more segments in different sequences. Touch can be used in conjunction with pose. For instance, spinning Facet around the wrist with a segment touched could result in a different action from spinning with nothing pressed (Figure 5d).

FACET MANAGER

Pose and multi-segment touch provide basic interaction components for Facet; however, we also provide a higher-level system that manages the mapping between applications and individual segments with the Facet Manager. An affordance of Facet's individual segments is enabling a one-to-one mapping between segment and content as illustrated in our scenario above. However, we want to allow the user to temporarily break this mapping to overcome the limited interaction space of individual segments. For example, when reading email the user might want to temporarily expand the application to an adjacent segment to have more space to read a message. Akin to a desktop operating system's window manager, the Facet Manager provides generalized capabilities for moving and resizing applications running on different segments.

At the core of the Facet Manager is a mechanism for mapping applications to individual segments and specifying how the segments work together. It provides a way for applications—at the user's request—to grow to encompass more segments, and to revert back to a single segment. The Facet Manager allows users to reorder applications on the bracelet without physically moving the segments. It also enables virtual segments—segments that are logically part of the circular configuration of Facet yet do not have associated physical segments. Although there are many possibilities for touch interaction (both on-screen and movement-based) to control the Facet Manager, we decided to use just three: a two-segment pinch, a two-segment "rotate", and a three-segment chord. This allows any touch events occurring on a single segment, and any other multi-segment touches, to be passed through to applications. Furthermore, there are potentially many different interactions that could be used to perform these operations; although we believe our selection make sense, there is room to investigate more optimal configurations.

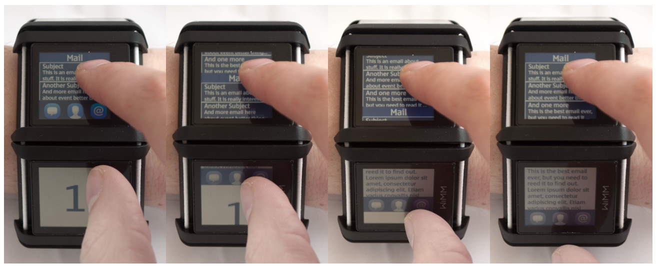

Figure 6: The *expand* operation. The user touches first on the top to indicate which segment to expand, then drags in the segment to which the application should expand. In this case, the application uses the extra segment to show more text. (View right-to-left for the *collapse* operation).

In addition to providing the above capabilities for Facet, the Facet Manager serves as a way to validate the overall system and interaction components. In particular, we use several of the interaction components in the context of a system with key functional capabilities.

Expand

Facet Manager allows the user to annex additional segments for an application. This *expand* operation is performed with a multi-segment touch reverse pinch (Figure 6). The segment of the first touch specifies which application to expand, while the second touch indicates which segment to expand onto. As the user expands the pinch, the new content from the application slides onto the second segment. Once the user slides the second finger past a threshold (currently 2/3 of the screen), the fingers can be lifted and the expansion completes. Lifting before the threshold is reached cancels the operation, reverting the system to its previous state. Expansion need not be just to adjacent segments; the user can skip intervening segments depending on their needs and the application's capabilities (Figure 5b).

Content shown on the new segment is application-dependent. One application might show more content, while another might show a keyboard or a menu. Additionally, the application can respond differently based on the initial touch points: when an expand is initiated, the Facet Manager provides the starting point to the application, which can perform a *contextual expand* based on the content touched (Figure 7).

When expanding, the new content displaces content on the affected segments. The visual effect of this operation is to

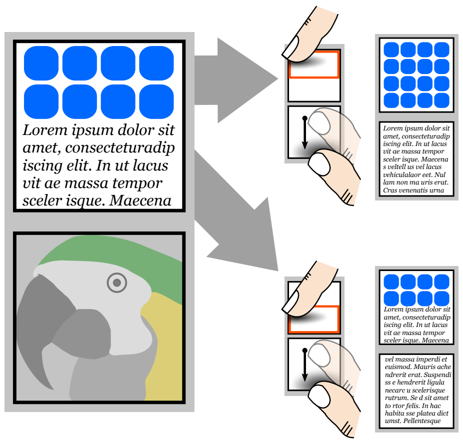
Figure 7: Two options for context-based expansion with the initial state (left), the orange rectangles showing where the user starts a touch (middle), and the results (right).

push all of the content around the bracelet by one segment. With the addition of the new application screen, the system now has more logical segments than physical devices it can show creating *virtual segments*. For a short interaction with a piece of expanded content, this mismatch is not likely to be an issue; however, in some cases the user may wish to see the hidden information. We include one mechanism (discussed below) for easily accessing the virtual segments.

Collapse
The logical inverse of *expand* is *collapse*, allowing the user to shrink an application taking up multiple segments. The collapse interaction is the opposite of expand (Figure 6, viewed right-to-left). As with expand, applications can respond to collapse differently based on context: if an application is using three segments, collapsing the first and second might have a different effect than collapsing the second and third.

Swap
The contents of two segments can be exchanged with the *swap* operation, which uses a touch interaction similar to the "rotate" gesture on larger multi-touch systems (Figure 8). With Facet, the two fingers used for the touch specify which segments to swap.

Expand All
While the expand operation can be used repeatedly to grow an application to multiple segments, there may be times when the user wishes to expand an application to all of the segments, allowing for interaction with a single application for a period of time. The *expand all* operation uses a three-finger chord (Figure 5c), with the first finger to touch indicating which segment to expand. The other two fingers are placed on the directly adjoining segments; when any finger is lifted, the application receives the expand all event and gains access to all of the segments. In some cases, an application may have fewer or more screens to show than there are segments. If fewer, the application expands to its maximum state, and the remaining segments show their original content. If an

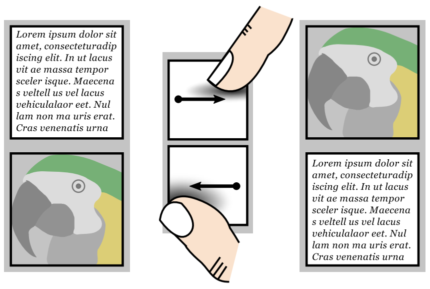
Figure 8: *Swap* exchanges the contents of two segments.

application has more content than available segments, it creates *virtual segments*, which can be accessed with the rotation mechanism described below.

Collapse All
To reverse the expand all, or collapse one or more segments that have been expanded, the user can employ the *collapse all* operation, which is simply a rapid back and forth wrist rotation, reminiscent of re-seating a traditional watch on one's wrist. This operation uses wrist angle data provided by the pose subsystem, looking for a rapid sequence of positive and negative changes in angle. To minimize false positives, the collapse all operation is only triggered when Facet is likely in use with the arm approximately horizontal ($\pm 30°$, as detected by the pose subsystem).

Virtual Segments
The Facet Manager support operations such as expand and expand all that may cause applications to occupy more segments than are physically available, resulting in hidden, or *virtual* segments. In order to view the virtual segments, the user can rotate the bracelet about the wrist. As it rotates, Facet shows all of the application screens in order. With this capability the user could, for example, rotate through nine virtual screens on six physical segments, with all of the content appearing in order (Figure 9).

To accomplish this effect, we create a discontinuity between segments that are difficult to see when in normal use. We use the pose subsystem to find the bottom segment, and the adjacent segment facing away from the user (segments D and E in Figure 5). Between these two segments is the discontinuity which is updated as Facet is rotated, so the user always sees the correct set of continuous segments. To see all of the segments, the user must rotate Facet more than 360°; for example, with nine segments, the user would rotate Facet 540°. Although not implemented currently, we have considered an "infinite" mode that would continually update as the user rotates; such a mode might be useful for long textual content or other widgets utilizing the continuous nature of the system.

DISCUSSION
The design and implementation of Facet allowed us to explore a number of issues and opportunities raised by its unique form factor. Facet's automatic pose calculation capability, coupled with multi-segment touch enabled us to build the Facet Manager, which in turn provides a platform for managing applications. We have shown how Facet allows applications to use

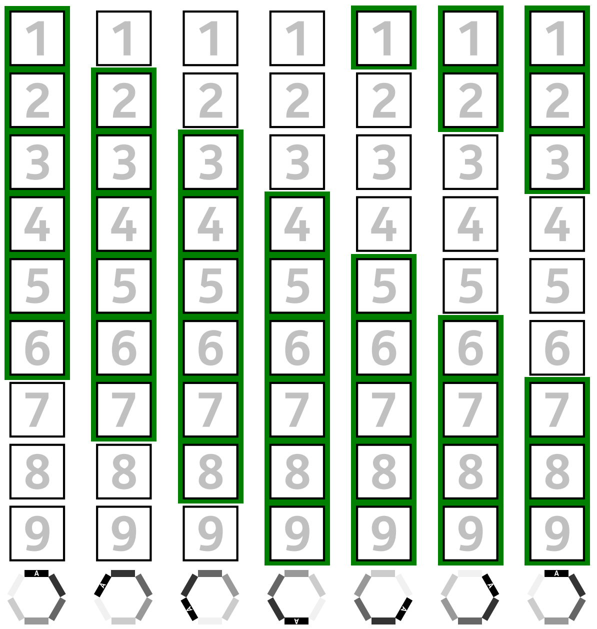

Figure 9: Progression of virtual segments. The dark green box indicates which virtual segments are visible, while the side view of Facet at the bottom shows its pose.

additional space, move between segments, and show more content than fits on the available segments. Facet in its current form is very much a prototype, but many of the lessons we have learned in its creation can be generalized.

From the beginning, we conceptualized Facet as a collection of identical segments that can be attached and removed by the user. If a device similar to Facet *was* to be constructed without removable segments some afforances would be lost, but much of the redundancy in hardware could be eliminated. It would require multiple touch screens to retain the segmented nature of the device, but the processing, storage, radios, and sensors could be consolidated. Most of the Facet Manager's interaction language could be retained: expansion, contraction, swapping and virtual segments would all still work. Taking the integration a step farther, one could imagine using a single curved touch screen wrapped around the user's wrist. In this configuration, the user would further lose the landmarks provided by segment boundaries which could result in different usage patterns as found in the use of multiple larger displays [7]. Explicitly comparing the relative affordances of these hardware configurations would be a useful area for future research.

Another design possibility would be to use heterogenously-sized hardware: it might be useful to have one segment larger than the others—analogous to a traditional watch face—with the others comprising a "band". Segments of differing aspect ratios—short and wide versus narrow and tall—could lend themselves to different application uses and touch affordances. A different point in a heterogeneous design space could be offering segments that are touch-only with no display, display-only with no touch, or segments with different sensing capabilities. The segment on the bottom of the wrist might be touch-only and offer indirect input as suggested by Baudisch and Chu's nanoTouch [1]. Depending on configuration, devices would lose some of Facet's application management and ability to interchange segments on an as needed basis but might gain in simplicity of use or cost.

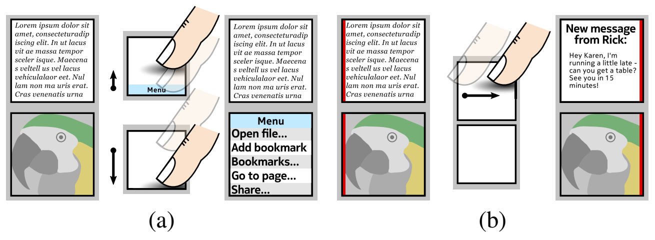

Figure 10: Proposed extensions to Facet. Image (a) shows menuing where a slight slide upwards from the bottom "pulls up" a menu; then a downward slide "pushes" the menu to the next segment. For notifications (b), a bar appears on the side of all segments. The user slides a finger from the side, and the notification appears on the segment where the touch occurred.

A unique aspect of Facet is multi-segment touch. Superficially, this is similar to multi-touch on other devices. However, Facet's touch functionality has unique properties. Due to its segmented nature, the user can distinguish between segments by touch alone, similar to a touch-typist's ability to feel the keys on a keyboard. In addition to the physical affordance of the touch-sensitive segments, Facet offers parameterization for input: the *pose* of the segment being touched can be significant even if the touches are otherwise the same.

FUTURE WORK

Our Facet prototype provides an excellent jumping-off point for future work. As a platform with unique affordances, there is an opportunity to develop additional interaction techniques. For example, a "slide up then down" gesture could summon a menu that is hidden at the bottom of a segment, using similar semantics to the *expand* operation (Figure 10a). Facet could receive and show notifications, but what if the relevant application segment is not visible to the user? One solution is to show the notification on *all* segments (Figure 10b); the user could then choose which segment to view the notification on.

Facet could be combined with other devices in a complementary way. For example, a phone's orientation sensor could be compared with each segment of Facet and detect which segment the phone has been tapped against. This would then pair that segment with the phone, providing an expanded application view, system configuration, or a larger input surface.

When wearing Facet on the wrist, visibility varies from segment to segment: those towards to the top and inward side (A–C in Figure 5) are more private; segments on the outside (E and F) are more visible to others; and the bottom segment (D) is difficult for anyone to see. Applications might leverage these varying levels of privacy: a social application might just show photos when publicly-visible, but display updates in the "private" position. Applications might also support collaboration by mirroring content among segments.

Facet could be used in other ways besides on the wrist: it can be held in the hand and rotated, collapsed flat to a double-sided three-segment-long display, or unbuckled and laid out flat in a row. These modes of operation could yield other interesting affordances to explore.

CONCLUSION

In this paper we presented Facet, a wearable segmented multi-display system. We described the technical platform as well as unique interaction components afforded by the platform and interaction techniques supported by the Facet Manager.

Facet simultaneously provides the capabilities of individual independent segments and of a cohesive collection. By taking advantage of the physicality of individual components, segments can be customized—manipulated as individual devices to run different applications in a wearable form factor with unique affordances.

Furthermore, Facet facilitates bringing those individual components together as a cohesive whole. The bracelet form factor enables a user to put several segments together and physically manipulate them as one device through pose detection and multi-segment touch, unique affordances that could be applied elsewhere. The pose detection further enables cohesion by managing the relationship between segments.

Finally, the Facet Manager allows the user to manage both the individual and collective nature of the system. The user can expand, move, collapse, and swap applications as needed. Virtual segments provide a mechanism for moving beyond the fixed set of segments by leveraging the continuous circular nature of Facet. Based on this work, we believe Facet points to a novel class of wearable device.

REFERENCES

1. Baudisch, P., and Chu, G. Back-of-device interaction allows creating very small touch devices. In *Proc. of the 27th international conference on Human factors in computing systems*, CHI '09 (2009), 1923–1932.
2. Blasko, G., and Feiner, S. An interaction system for watch computers using tactile guidance and bidirectional segmented strokes. In *Proceedings of the Eighth International Symposium on Wearable Computers*, ISWC '04 (2004), 120–123.
3. Blasko, G., and Feiner, S. Evaluation of an eyes-free cursorless numeric entry system for wearable computers. In *Proc. of the 10th International Symposium on Wearable Computers*, ISWC '06 (2006), 21–28.
4. Blasko, G., Narayanaswami, C., and Feiner, S. Prototyping retractable string-based interaction techniques for dual-display mobile devices. In *Proceedings of the SIGCHI conference on Human Factors in computing systems*, CHI '06 (2006), 369–372.
5. Butz, A., Groß, M., and Krüger, A. Tuister: a tangible ui for hierarchical structures. In *Proceedings of the 9th international conference on Intelligent user interfaces*, IUI '04, ACM (New York, NY, USA, 2004), 223–225.
6. Crossan, A., Williamson, J., Brewster, S., and Murray-Smith, R. Wrist rotation for interaction in mobile contexts. In *Proceedings of the 10th international conference on Human computer interaction with mobile devices and services*, MobileHCI '08 (2008), 435–438.
7. Grudin, J. Partitioning digital worlds: focal and peripheral awareness in multiple monitor use. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '01 (2001), 458–465.
8. Hansson, R., and Ljungstrand, P. The reminder bracelet: subtle notification cues for mobile devices. In *CHI '00 extended abstracts on Human factors in computing systems*, CHI EA '00 (2000), 323–324.
9. Harrison, B. L., Fishkin, K. P., Gujar, A., Mochon, C., and Want, R. Squeeze me, hold me, tilt me! an exploration of manipulative user interfaces. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '98 (1998), 17–24.
10. Harrison, C., and Hudson, S. E. Abracadabra: wireless, high-precision, and unpowered finger input for very small mobile devices. In *Proceedings of the 22nd annual ACM symposium on User interface software and technology*, UIST '09 (2009), 121–124.
11. Hinckley, K. Synchronous gestures for multiple persons and computers. In *Proceedings of the 16th annual ACM symposium on User interface software and technology*, UIST '03 (2003), 149–158.
12. Hinckley, K., Dixon, M., Sarin, R., Guimbretiere, F., and Balakrishnan, R. Codex: a dual screen tablet computer. In *Proc. of the 27th international conference on Human factors in computing systems*, CHI '09 (2009), 1933–1942.
13. Hinckley, K., Pierce, J., Sinclair, M., and Horvitz, E. Sensing techniques for mobile interaction. In *Proc. of the 13th annual ACM symposium on User interface software and technology*, UIST '00 (2000), 91–100.
14. Holmquist, L. E., Mattern, F., Schiele, B., Alahuhta, P., Beigl, M., and Gellersen, H.-W. Smart-its friends: A technique for users to easily establish connections between smart artefacts. In *Proceedings of the 3rd international conference on Ubiquitous Computing* (2001), 116–122.
15. Hutterer, P., Smith, M. T., Thomas, B. H., Piekarski, W., and Ankcorn, J. Lightweight user interfaces for watch based displays. In *Proceedings of the Sixth Australasian conference on User interface - Volume 40*, AUIC '05 (2005), 89–98.
16. Kim, J., He, J., Lyons, K., and Starner, T. The gesture watch: A wireless contact-free gesture based wrist interface. In *Proceedings of the 2007 11th IEEE International Symposium on Wearable Computers*, ISWC '07 (2007), 1–8.
17. Lucero, A., Keränen, J., and Korhonen, H. Collaborative use of mobile phones for brainstorming. In *Proceedings of the 12th international conference on Human computer interaction with mobile devices and services*, MobileHCI '10 (2010), 337–340.
18. Lyons, K., Pering, T., Rosario, B., Sud, S., and Want, R. Multi-display composition: Supporting display sharing for collocated mobile devices. In *Proceedings of the 12th IFIP TC 13 International Conference on Human-Computer Interaction*, INTERACT '09 (2009), 758–771.
19. Martin, T. Time and time again: Parallels in the development of the watch and the wearable computer. In *Proc. of the 6th IEEE International Symp. on Wearable Computers*, ISWC '02 (2002), 5–11.
20. Merrill, D., Kalanithi, J., and Maes, P. Siftables: towards sensor network user interfaces. In *Proc. of the 1st international conference on Tangible and embedded interaction*, TEI '07 (2007), 75–78.
21. Poupyrev, I., Newton-Dunn, H., and Bau, O. D20: interaction with multifaceted display devices. In *CHI '06 extended abstracts on Human factors in computing systems*, CHI EA '06, ACM (New York, NY, USA, 2006), 1241–1246.
22. Raghunath, M. T., and Narayanaswami, C. User interfaces for applications on a wrist watch. *Personal Ubiquitous Computing 6* (January 2002), 17–30.
23. Rekimoto, J. Tilting operations for small screen interfaces. In *Proceedings of the 9th annual ACM symposium on User interface software and technology*, UIST '96 (1996), 167–168.
24. Rekimoto, J. Gesturewrist and gesturepad: Unobtrusive wearable interaction devices. In *Proceedings of ISWC '01* (2001), 21–27.
25. Vogel, D., and Baudisch, P. Shift: a technique for operating pen-based interfaces using touch. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '07 (2007), 657–666.
26. Williams, A., Farnham, S., and Counts, S. Exploring wearable ambient displays for social awareness. In *CHI '06 extended abstracts on Human factors in computing systems*, CHI EA '06 (2006), 1529–1534.
27. Yatani, K., Partridge, K., Bern, M., and Newman, M. W. Escape: a target selection technique using visually-cued gestures. In *Proceedings of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, CHI '08 (2008), 285–294.