COMMAND AND CONTROL IN A MULTITOUCH ENVIRONMENT

R. Szymanski (US Army), M. Goldin (US Army), N. Palmer (US Army),
R. Beckinger (US Army), J. Gilday (US Army), T. Chase (Denim Computer Systems, Inc.)
AMSRD-CER-C2-BC
Fort Monmouth, New Jersey 07703

INTRODUCTION

As the Army continues its movement toward Net-Centricity, the need for face-to-face collaboration remains. Although commanders will eventually have the ability to utilize large bandwidth pipes and robust software services to collaborate remotely, there is no indication commanders will loose their desire to conduct localized planning sessions "eyeball-to-eyeball." However, little is currently being done to provide computer assistance for in-person collaboration. This paper discusses the use of advanced user interface technology to create a computer enhanced multitouch command and control collaborative environment. The COMET (Command and Control Multitouch Enabled Table) team out of CERDEC C2D (Communications Electronics Research Development and Engineering Center Command and Control Directorate) is building and researching the system discussed in this paper.

1.1 The Need for Face-to-Face Collaboration

Reports in the form of lessons learned and Battle Command application usability studies suggest that commanders still have a strong desire to conduct local analog collaboration in addition to distributed digital collaboration. COL (P) Bayer, 3RD Infantry Division noted,

"I would literally pull a map off the wall … I'd give him a quick briefing about what was going on in the division … Decisions were made by commander looking at a subordinate commander eyeball to eyeball rather than relying primarily on the science piece that's gathered through technology and fed to staffs" [1]

Traditional "analog" collaboration is implemented with paper maps, grease pens, acetate layers, and sand tables. While these tools are intuitive and can work exceptionally well to get one's point across, they are ineffective at recording, saving, and transmitting information. This is especially true when the information needs to feed ancillary computerized decision support systems. In addition, traditional tools are incapable of providing automated assistance or analysis. Our initial research indicates that a useful face to face digital collaboration tool is one that enables users to interact with the system as if it were analog. A digital system enabling analog collaboration offers the best of both words.

LTG Wallace provides important additional insight into the need for face to face collaboration by observing

"No matter how sophisticated the technology may become in providing a seemingly improved picture of the battlefield, the true 'centre' of effective command and control (C2) remains the commander" [2]

Army computer scientists have been focusing on developing computer systems that provide advanced functions for the commander and his staff. However, they often fail to realize that the human should always be at the center of the system. COL (P) Bayer stressed that often too much emphasis is placed on "Net-Centric Battle Command" and not enough is placed on "Commander-Centric, Network-Enabled Battle Command." Based on our research it is our conclusion that multitouch computer systems can improve commander centricity.

1.2 Existing Multitouch Environments

Perhaps today's most ubiquitous multitouch interface is the Apple iPhone and systems have been developed by Microsoft, HP, Perceptive Pixel, and others. Most people who come in contact with the iPhone can learn to use its features within a few minutes without the aid of a user's manual. This is because "things behave as you'd expect." An example is the map viewing interface: you drag your finger across the map to pan, "pinch" it with two fingers to zoom out, "stretch" it with two fingers to zoom in. There are no menus to cycle through, no "right clicks", and no keystrokes to memorize. To strengthen the sense of reality, objects behave as expected when they move by exhibiting momentum and friction. The "wall" between the user and the computer seems to disappear.

1.3 COMET Work

The COMET team built a multitouch environment to use as a foundation for the development of multitouch enabled C2 applications. While there are many COTS (Commercial Off-The-Shelf) multitouch hardware and software solutions[3,4,5], we decided to build our own from the ground up so we could better understand how an entire system could operate, obtain a clearer view into the inner

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
01 DEC 2008	**N/A**	**-**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
Command And Control In A Multitouch Environment	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **AMSRD-CER-C2-BC Fort Monmouth, New Jersey 07703**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release, distribution unlimited

13. SUPPLEMENTARY NOTES
See also ADM002187. Proceedings of the Army Science Conference (26th) Held in Orlando, Florida on 1-4 December 2008, The original document contains color images.

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			
unclassified	**unclassified**	**unclassified**	**UU**	**7**	

workings of the technology, and to ultimately build better and more finely tuned applications.

There are several popular multitouch hardware technologies to choose from but, because of its relative simplicity, we built our hardware based on Frustrated Total Internal Reflection (FTIR). Perceptive Pixel's Jeff Han[6] was one of the first to demonstrate a full-featured FTIR based multitouch system. Given the large amount of information on the Internet explaining how to build FTIR devices[7,8,9], we will only briefly explain the hardware here and focus instead on multitouch-enabled C2 applications.

1.4 A Brief Introduction to FTIR

The COMET hardware is based on the total internal reflection (TIR) of light within a medium. TIR occurs when light such as that from a light emitting diode (LED) is introduced along the edge of a piece of clear acrylic. When the LEDs are correctly positioned, most of the light will travel within the acrylic reflecting back and forth between the top and bottom of the plastic sheet due to the different indexes of refraction of the acrylic and the surrounding air. If you alter the index of refraction on either side of the acrylic by touching the surface you can cause the internal reflection of the light to become "frustrated" so that the light scatters perpendicular to the the surface. For multitouch applications the touching object is usually a finger (or a pen/stylus, or a cup of coffee, etc.). This is illustrated in Figure 1.

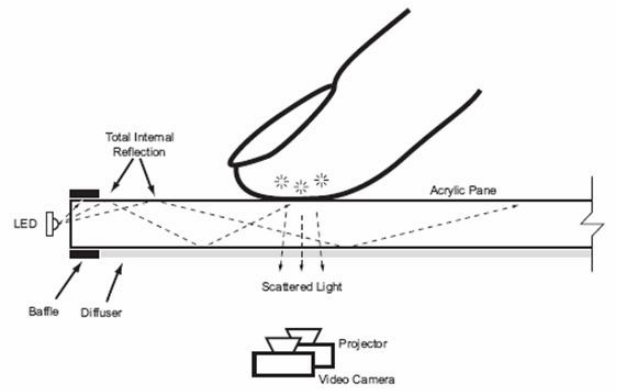

Figure 1: Jeff Han's depiction of how FTIR works (ref 6)

An infrared camera is placed below the acrylic and is able to see the scattered light as distinct finger presses, or *blobs*. We used infrared light and infrared filters on the camera so that the camera would only see the scattered light and not pick up any ambient visible light. When users touch the surface the video camera sees the resulting touches as white blobs against a black background as shown in Figure 2. In addition to the camera we positioned a standard video projector below the acrylic surface to provide the visual display to the user. The projector projects its image onto a translucent surface mounted below the acrylic. We used tracing paper because we found that it could display the projected images yet would pass the infrared light from the finger touches.

Software based on computer vision techniques maps the blobs' camera coordinates into the projector coordinates taking into account keystone effects and surface, camera, and projector alignment issues. We used affine transformation matrices calculated from a 4-point alignment procedure. The net result is a programming environment that provides touch events for multitouch-enabled applications. Touch events are similar to mouse events in conventional single user applications except with multitouch there are conceptually any number of simultaneous "mice" each of which is associated with a recognized blob. The events COMET recognizes are *blob down* when a touch begins, *blob move* when you drag your finger over the surface, and *blob up* when you stop touching the surface. At any given time there can be any number of active blobs with each blob given a unique identifier.

Figure 2: Infrared image of 5 blobs (4 fingers + thumb)

The simple blob events can be augmented by introducing the concept of *gestures*. Gestures are blob movement patterns representing actions that extend the blob event vocabulary to provide additional semantics for application use. For example, a blob moving in a circle could invoke a menu while two blobs moving apart could mean stretch an object. Our software was developed using Microsoft's .NET C# programming language. In order to achieve the desired fluid user interface effects we used the Windows Presentation Foundation (WPF) user interface environment developed for the Vista operating system. For more details please consult the references at the end of this paper.

1.5 Designing C2 Multitouch Applications

Using the COMET multitouch platform the team started designing multitouch applications. We evolved two different approaches to application development. The first was to retrofit mouse based applications (e.g. Google Earth) to work in a multitouch environment while the second approach was to design and build multitouch

applications from scratch. Our experience shows that converting mouse based applications to work in a multitouch environment often does not produce satisfying results. Such applications are typically developed for a single user and with a keyboard and mouse in mind. As a result, the introduction of a multitouch front end often produces odd effects. This is especially true if multiple users interact with the same application at once. One can map gestures into application macros, for example, a stretching motion could be mapped to Control-Z to zoom out a map. However, none of the solutions that we've encountered seems as fluid as an interface specifically designed to support a multitouch, multiuser environment. Results obtained by single users interacting with multitouch enabled front ends added to mouse based applications can be quite good, but we found it difficult to extend this success to multiple simultaneous users accessing the same application.

Based on our command and control interests we felt a good multitouch application would be a digital sand table. We use the term "sand table" a bit loosely here, what we mean is any non-digital planning tool. This could literally be a sand table or it could be a paper map and acetate. One reason commanders seem to still like sand tables is that users can directly interact with the planning surface; there is no mouse, no keyboard, and no pen. A mouse is a layer of abstraction that does not allow the user to directly manipulate the objects that he is creating. It is our belief that the closer the user is to the object he wishes to manipulate, the easier it is for him to use the system. Our investigation shows that tactile interfaces modeling the physical world are easier to understand and use. Picking up a unit and moving it on the map is a natural act requiring little training to understand.

The fundamental flaw with "analog" sand tables in the digital age is that there is no easy way to retrieve information generated with the table or to share that information with remote team members. Further, sand tables can't assist the commander with planning analysis tasks such as calculating how long it will take to move from one location to another or how much fuel will be used in the move. The COMET team proposes to solve these problems by creating a multitouch digital sand table. This is a digital representation of a sand table allowing multiple users to interact at the same time. Because it is digital, it can store, playback, jump through time, compare, and reason. The table can be used throughout the mission's lifecycle during planning, as an interactive COP display, and for replanning operations. Additionally, it can transmit digital information about the plan to remote team members or enable remote users to collaborate on the mission.

The most significant difference between a digital sand table running on COMET and any other planning system is that users can directly manipulate the data. They can draw on the map with their fingers, they can "pick up" objects and move them, and they can pan, zoom, and rotate the map with their hands all while collaborating eyeball to eyeball across the work surface.

Figure 3: COMET in Use in TOC at Fort Dix

This is as close as one can get to directly manipulating a map without actually having a real physical map. Further the resulting computer-driven maps can be much more powerful than real physical maps and provide properties that are impossible with paper maps. For example, touching an area on the map can allow a user to drill down into geo-referenced information associated with the map at the point of touch. Figure 3 shows COMET being used at the PM Command, Control, Communication, Computers, Intelligence, Surveillance and Reconnaissance On The Move 2008 demonstration event at Fort Dix New Jersey.

1.6 Surface Management

An approach to the management of the multitouch surface is central to multitouch application development. We observed that there were two fundamentally different approaches to surface management. The first is to develop what would amount to a multitouch extended operating system (MOS). Such a system would enable multiple individual applications to occupy the surface simultaneously with the MOS dispatching touch events and gestures to the different applications on the surface. This apparently what Microsoft is doing with Windows 7. The second surface management approach is to write a single application that takes control of the entire surface, receives all touch events and manages all aspects of the application. The single application approach is by far the simpler of the two so we elected to build a full-surface application that would be the only one executing. Such applications do, however, perform many different tasks at once and need to be able to simultaneously manage multiple windows on the display surface.

In order to foster small group eye-ball-to-eyeball collaboration we chose a horizontal "coffee table" environment similar to Microsoft Surface[10]. As a result of the screen's orientation the usual notions of top, bottom, left, and right are no longer meaningful. Because users position themselves around the screen, having a traditional "Start" menu at the bottom-left of the interface doesn't allow users at the opposite sides of the screen to easily access tools. For our first application we decided to add toolbars to two opposing sides of the screen. Experience with this approach has shown that it is not an optimal solution. Ideally users should be able access the tools from anywhere on the screen. A better solution would be a gesture that displays a toolbar anywhere the user wants. This approach has been used effectively by Perceptive Pixel's system.[5]

1.6.1 COMET Application Functions

Based on these concepts, the COMET team designed an application that demonstrates the potential for multi-touch, multiuser, commander centric, network-enabled collaboration environment. The demonstration application we developed was immodestly dubbed SlickDemo. The SlickDemo application supports the following functions:

1. Display multiple maps and planning information
2. Create a drawing surfaces that allows users to draw basic graphics on any object
3. Show live streaming video from Unmanned Aerial Systems (UASs)
4. Show static imagery files

Figure 4: Typical COMET Desktop Display

Although not a full-fledged C2 application, SlickDemo's collection of features gave users a sense of how multitouch technology could be used to access basic planning and Intelligence tools. In addition SlickDemo provided a useful experimental platform enabling us to gather user comments and suggestions. A screen capture of the application is shown in Figure 4. The actual screen size is about 2 feet by 3 feet.

The figure illustrates the menu bars on the top and bottom of the display. These are the small icons such as the house and recycle bin. The large rectangular areas on the left and right are frame distribution controls. Users can drag frames from them to the display. The frames can contain images, videos, and maps. The large object in the center of the display is an image. The orange circles mark user touches.

1.6.2 COMET Application Look and Feel

Immediately we saw that users gravitate toward on-screen objects that have familiar behaviors so we developed a set of interface guidelines to help insure consistent object behavior. For example, no user action should result in a sudden appearance, disappearance, or change of a GUI object. A sudden change in the GUI forces the user to perceive the change by visually comparing the before and after states of the GUI. Real objects just don't behave with such abruptness. Therefore any change in an object's existence or layout uses one or more synchronized graphical animations in order to illustrate its transition. A good example of this is when a user touches a button to create a picture object on the workspace: the button displays a quick flash animation to give feedback that it has been touched and rather than just appearing, the created picture flies out from under the user's finger by stimulatingly animating translating and scaling transforms to move it to the center of the workspace. Similarly, when a user applies a command to hide a set of buttons, the buttons fade away to illustrate their disappearance.

We have discovered that in a multi-user environment, where changes to the GUI can be performed by anyone, animations are much more than just a novelty; they are an important aid to understanding. An animation that results from a user's direct interaction with an object helps the user track the change (s)he has made and distinguishes this change from other simultaneous changes caused by other users.

1.7 Problems and Observations

The team was fortunate enough to be able to display COMET at Fort Dix during the 2008 demonstration event. This gave us the opportunity to put the table in front of real warfighters and to become part of their planning workflow in the TOC. As a result of user feedback and our observations we discovered the following problem areas.

Cluttered desktop. While undergoing active use with several people accessing COMET at the same time, the desktop tends to become cluttered. "Where'd I put that UAV video?" was a typical comment. One solution that appeared promising was a de-clutter function that caused all of the desktop objects to arrange themselves in a meaningful order.

Ownership. Our system has no sense of ownership with respect to desktop resources. It is possible for user A to create an image and for user B to destroy the image. This is especially troubling when a user invokes the "clear the entire desktop" function and everyone's work is lost. It is important to develop some sense of ownership for the various objects on the desktop. How to show ownership can be a complex problem as discussed below under menus and borders.

Orientation. On a flat table top surface orientation can be a problem. For some applications this may not be an issue. For example, when examining a map or a still image from a UAV orientation may not be of any concern, but when reading a 5-paragraph OPORD, unless you can read English upside down, orientation is definitely an issue. We found our users often had to rotate images 180 degrees to show them properly to people on the "other" side of the table. In addition, we had to insure our software would launch newly created images with an initial orientation that matched the person who created the image. Eyeball-to-eyeball collaboration across a horizontal surface has these problems while the same touch surface hung on a wall does not because the wall mounted surface clearly as an up and a down that is in alignment with all users looking at the surface, but as discussed below both horizontal and vertical solutions have their individual strengths and weaknesses.

Touch context. We found that touches have to take on different contexts. For example, if you implement a simple drawing feature (e.g. dragging your finger over an image permits you to draw on the image), then the meaning of a finger drag could be either "draw where I'm pointing" or it could mean "I want to move the picture where I'm pointing." Our solution to this problem was to have windows maintain *state*. In the normal state a drawing program has its display moved when you move your finger over it or perform some other gesture. In the drawing state, however, the gestures are interpreted as drawing operations. There must be a clear indication of the state of the window however and as with displaying ownership this may not be that easy to do because of problems maintaining borders around windows (see following)

Menus and borders. The standard procedure of having menus or borders on various edges of windows can also be problematic. The COMET table, much like the Apple iPhone and other touch surfaces, permits users to zoom in on items by using the "two-finger spread" gesture. The difficulty is that by using the gesture it is easy to make an item grow so that its borders are off the screen. The zooming also moves drag handles and menus associated with the image off the screen as well. This can be a real problem when dealing with maps because it is natural to zoom a high-quality map in for street level details. Our solution, and in retrospect perhaps not a great one, was to have menus be separable from the images they are associated with. This permits a user to position the menu wherever (s)he wishes letting the associated image zoom to any size without fear of menu loss. This solution has its own problems because the image and the menu are physically separated and on a cluttered screen the notion of which menu belongs to which image may become confused. Our solution was to provide a "locate button" that draws a "laser beam" from the menu to its associated window. An even better solution might be to use a gesture on any given window to pop up a context menu.

Gesture overload. We found gestures to be powerful features as long as the gesture is "natural." The problem with gestures as opposed to continuously visible menu options is that gestures must be learned and remembered while menus are always visible and can be used as their own mnemonic. Some gestures are almost innate in humans because they are either duplicative or reminiscent of actions performed on physical objects. We found the act of dragging a window to be totally natural, but we did have to put special code in to permit "multi-finger" dragging because people would use three or four fingers at the same time to move an object perhaps in an attempt to simulate actually picking the object up.

The zoom in and out gesture formed by moving two fingers apart (zoom in) or together (zoom out) is also natural because it is similar to the movements required to stretch or shrink a rubber-like sheet. Beyond that small collection, however, gestures can become complicated and difficult to remember. A universal "show me the menu" gesture would probably have been a good idea, but we opted instead for permanent on-screen menus. Further gestures need to be taken in the context of the object being manipulated. A zoom gesture on an object that cannot zoom makes little sense. We found the following general gesture heuristics to be useful:

- Make gestures consistent with the physical manipulation of the object being shown. For example, to spin a wheel-like display you should use a spinning gesture.
- Make gestures as consistent as possible across all screen objects. In other words if gesture "A" means something with an object of type 1, then ideally it should mean the same thing with an object of type 2.
- Keep the gesture set small so users don't have to learn what could easily become a new style of writing.

Sometimes touch isn't the answer. Touch can be a powerful model for interaction with a computer but we found that it isn't the be all and end all. Typing, for example, is clearly best done with a keyboard. Although COMET doesn't yet have keyboards attached we could clearly envision wireless keyboards that would be used to type into documents on the surface. We experimented with virtual keyboards and found them tedious for anything except the most rudimentary data input. Any touch typist would be quickly driven insane if required to use an on-surface virtual keyboard. The artistry with touch is to be able to know when to use the surface and when to use some other interactive mode. The multitouch surface should be considered one more man/machine path in a multi-modal interaction scheme.

Sometimes a table isn't the answer. Our initial concept was built as a horizontal surface. Feedback from users has led us to believe a convertible surface might be more useful. This would be a device that could easily switch between a vertical (whiteboard style) and a horizontal (coffee table style) layout or anything in between. Users commented that while a horizontal interface works well for some collaborative tasks, a vertical interface would be better for others. After watching users we'd have to agree. There appears to be three different types of group collaborations that take place. We've named them:

- *Group search*. A small group of people work collaboratively to find an acceptable sequence of operations or sets of features. This type of collaboration includes things like mission planning, wargaming, assessing a common operating picture, or intel analysis. We believe group searching is best performed on a horizontal multitouch surface.

- *Discussion presentation*. A larger group is looking at a presentation, but there is discussion and changes to the display from a number of members of the group. This is the mode of operation that typically takes place at a white board between several individuals with an audience of on-lookers. This operation is probably best performed on a vertical multitouch surface where several people can engage one another at the board while others look on.

- *Briefing*. This is another vertical surface where one person explains something to a group in a classroom-like teaching experience. The touch surface can help provide an effective brief, but multitouch is probably not required for this type of interaction.

Our research has shown that an additional benefit from horizontally oriented touch tables is that the tables may be more "arm friendly." Persistent use of vertically oriented touch screen can produce arm discomfort. This condition, coined *gorilla arm* can make vertically oriented touch-screens uncomfortable to use for any significant period of time.

Height of the table. When people first encounter the COMET surface they often ask "Gee why is the thing so darn tall?" The top of the table is a full 52 inches above the floor. This is so tall, in fact, that we had to build a one foot high platform around the surface so that people could more easily interact with it. The reason for the height was that in order to get a two foot by three foot surface we had to have our blob spotting camera 52 inches from the surface in order to "see" the entire table top. A better design would feature multiple cameras with each viewing a portion of the table top. Multiple cameras, though, requires that the software be able to handle multiple image streams and at some level fuse the streams together. Our initial attempts to do this suggest that the fusing be done after blobs are recognized rather than at the video pixel level (there are *many* more pixels to fuse than there are blobs to fuse). Microsoft's Surface Computer is reported to use four blob recognition cameras.

Sand. The top of the table is made from acrylic plexiglas. This has a number of useful features but being scratch resistant isn't one of them. It is an odd fact of life that a digital sand table apparently abhors sand. To be truly useful in a TOC a multitouch system must have a surface that isn't easily damaged by sand. Alternatively there could be a surface cover that can be easily replaced when it is damaged.

User identification. We don't know how to recognize individual users when they touch the table, but it would very useful to be able to. Several solutions have been proposed. One is to wear special gloves that allow individual users to be recognized but one of our goals was to not require special styluses or other pointing equipment. Another suggestion was to use computer vision from above to correlate with the blob recognition under the screen. User recognition is still an unsolved problem worthy of additional research.

1.8 Other ideas

The multitouch planning surface suggests an interesting assortment of potential uses in addition to sand tables. For example, one could imagine that the surface could be modified to also act as a scanner/digital imager. While scanning a document is not a novel concept, scanning combined with the interactive surface is innovative. You could take hand written field notes, scan them into the table, and then edit or make comments by circling sections with your finger and writing notes. Those notes could then be shared with other local and remote users.

Paper maps could eventually be replaced by a thin, malleable multi-touch enabled displays which the war-

fighter can fold or roll for use on the move. Commercial multi-touch screens are currently available in the form of tables, whiteboards, tablet PCs, and handheld devices. Due to physical size and a difference in touch-sensing technology, multi-touch enabled tablet PCs and handhelds are portable but lose the advantage of multi-user interaction. Multitouch enabled walls and tables have the advantage of running multi-user collaborative applications but, due to size, are most at home in a tactical operations center (TOC) at Company and above.

In the planning process, the warfighter should be able to use their multi-touch digital map to analyze an urban area of interest (AOI). The warfighter should be able to inspect the AOI in 3D from a street level perspective to identify possible threats. Interacting with 3D space is typically accomplished with a normal computer mouse or joystick. Multi-touch gestures remove the necessity for these tools by allowing users to directly interact with the display in a more intuitive manner.

During an operation, a multi-touch display could be networked to all available services and display live SA. The commander should be able to select blue force units on the display and view their SA data, view any live video streams the units may provide, and have the option to send new tasks to and communicate via voice or chat with the commanding officer of the units. The commander should be able to select red force units on the display and view any Intel associated with them. The commander should be able to task his or her unmanned assets directly on the COP.

1.9 Conclusion

This paper has described CERDEC C2D's ongoing efforts to explore using multitouch technology for Command and Control applications. This effort is driven by lessons learned and studies that have shown that while digital tools are useful, under certain circumstances, users still prefer to collaborate, plan, and share information face-to-face. Additionally, those digital tools should directly support the goals of the user and the user (not the digital tool) should remain at the center of the system. The tools that are discussed in this paper are an attempt at enhancing localized collaboration through the use of digital components. We conclude that through the use of multitouch interfaces, engineers can build C2 applications that are useful, innovative, easy to use, and require little to no training.

---

[1] Cameron, R.S., Thunder Run Interview of COL (P) Bayer 3ID G3, Aug 2007

[2] Wallace, W.S. LTG, "Network-Enabled Battle Command", Military Review, May-June 2005, http://usacac.army.mil/CAC/milreview/download/English/MayJun05/wallace.pdf

[3] http://multitouch.fi/ready-to-roll

[4] http://nortd.com/touchkit/index.html

[5] http://www.perceptivepixel.com

[6] Han, J. Y., "Low-Cost Multi-Touch Sensing through Frustrated Total Internal Reflection", Proceedings of the 18th Annual ACM Symposium on User Interface Software and Technology, 2005

[7] http://www.multitouch.nl/documents/multitouch display_howto_070523_v02.pdf

[8] http://www.nuigroup.com/forums/

[9] http://www.digitalstratum.com/programming/ftir_build

[10] http://www.microsoft.com/surface