

Synchronous Paired Collaboration Support with the Transparent Video Facetop

Authors
Affiliations
emails

Abstract

The Transparent Video Facetop is a novel user interface concept that supports not only single-user interactions with a PC, but also close pair collaborations, such as that found in collaborative Web browsing, remote medicine, and in distributed pair programming. Facetop seamlessly couples PC desktop content with a semi-transparent, full-screen, live video user self-image. We recently demonstrated our first Facetop prototype as a single-user GUI for manipulating the elements of a traditional WIMP desktop [12]. In this paper we present a two-headed collaborative version of the Facetop, and discuss its use in solving several problems reported to us by teams doing distributed pair programming. Specifically, the Facetop allows a distributed pair to recapture some of the facial expressions and face-to-face communications contact lost in earlier distributed sessions. It also allows members of a distributed pair to conveniently, quickly, and naturally point to their shared work, in the same manner (manually) that they do when seated side-by-side.

Distributed Pair Programming

Our Facetop system development has been driven by collaboration needs and problems that naturally arose in a collaborative software engineering technique called pair programming. In pair programming, two programmers sit at one PC to develop code. One types (“drives”) while the other reviews and assists (“navigates”); roles swap frequently. Pair programming is a central practice in several agile software development methods, including Extreme Programming (XP) [1,2].

The benefits of pair programming are well known in co-located situations [3,13]; we have been exploring if the benefits remain in a distributed context. It has been established that distance matters [14]; face-to-face pair programmers will most likely outperform distributed pair programmers in terms of sheer productivity. However, the inevitability of distributed work in industry and education calls for research in determining how to make this type of software development most effective.

In several previous experiments in distributed pair programming (dPP) [6,7,11], a pair worked synchronously and remotely using COTS sharing software. *pcAnywhere* (Symantec) provided a shared desktop, so that the two programmers are effectively working on a single host computer, and each sees exactly the same desktop as they would were they sitting side-by-side at the host PC. *Yahoo messenger* provided voice communication, and text exchange when needed. Others have tried dPP with VNC, NetMeeting, and the shared desktop in Windows XP.

Our experiments found that programmers working synchronously in this distributed environment were as effective as co-located pair programmers. In post-experimental interviews, teams consistently told us 3 things:

- They missed facial expressions and the sense of presence obtained in side-by-side interaction
- They wanted a way to point at the shared work they were discussing via the audio channel.
- They wanted a whiteboard for drawing and design work

To address these issues, we have been investigating a video-enhanced dPP environment. Video was one issue discussed at a workshop on distributed pair programming at the XP/AU 2002 conference. Over 30 people attended, many of who had tried some form of distributed pair programming and were working on tools to improve its effectiveness. The consensus on video was that “web cam” style, postage stamp video – small image and low frame rate – was of little value in enhancing communications or sense of presence in a distributed pairing. However, it was felt that video, if large enough and real enough, was of potential value and worth further research. We have been doing that research in that context since that time.

Other Related Work

Aside from agile development and pair programming, our Facetop work depends on technology from several research areas: from collaboration theory and systems, from video analysis, and from user interfaces. While space does not permit a full overview of related work here, we do give such a review in a recent technical report on Facetop [2]. Englebart demonstrated analog video for collaboration in 1968 in his NLS/Augment system. Transparency has been used in user interfaces for tools and menus, avatars, and other aspects of the GUI [8,10]. VideoWindow [5] showed that large video can create a sense of presence that will encourage remote user interaction.

The work most closely similar to Facetop is *Clearboard* [4], developed by Ishii et al. and published in CSCW in 1994. *Clearboard* presented visual images of collaborating users to each other, but it required expensive custom hardware. One of the advancements of Facetop is potentially ubiquity, since it needs only a \$100 Firewire camera to be added to a common PC. The novelty of Facetop is using graphics transparency to tightly integrate the user image with the desktop work. Toolkits such as *videoSpace* [15] are available for producing application like Facetop, though the heavy processing and transparency in Facetop make Apple development most efficient with current platforms.

The Basic Single-User Facetop

Transparency combined with user self-view

The transparent video Facetop is a novel enhancement of the traditional WIMP user interface. Figure 1 shows the physical Facetop setup for a computer with a monitor. Note the video camera sitting on top the LCD panel pointing back at the user; in

our current work we use a \$100 Sony iBot, giving us an image that is 640 x 480 pixels of 24-bit color, captured 30 frames per second. We take this user self-image video stream, reverse it horizontally, make it full screen, make it semi-transparent (done on a high-performance graphics card), and composite it with the desktop image. The user then sees him/her self as a “ghostly” image apparently behind the desktop, looking back at the icons and windows from the behind. Instead of a traditional desktop, we see a “face” top.

Reversing the image horizontally means that when the user moves a hand, say, to the left, the image of the hand mirrors this movement on the screen. Thus, if the user reaches out and points at an icon, say, the Facetop image points at the same icon. *Camera registration is not an issue in Facetop*, unlike many camera-based video systems. Since the user employs the video self-image to guide finger movement in pointing, we can even jostle or relocate the camera during use without loss of proper Facetop function

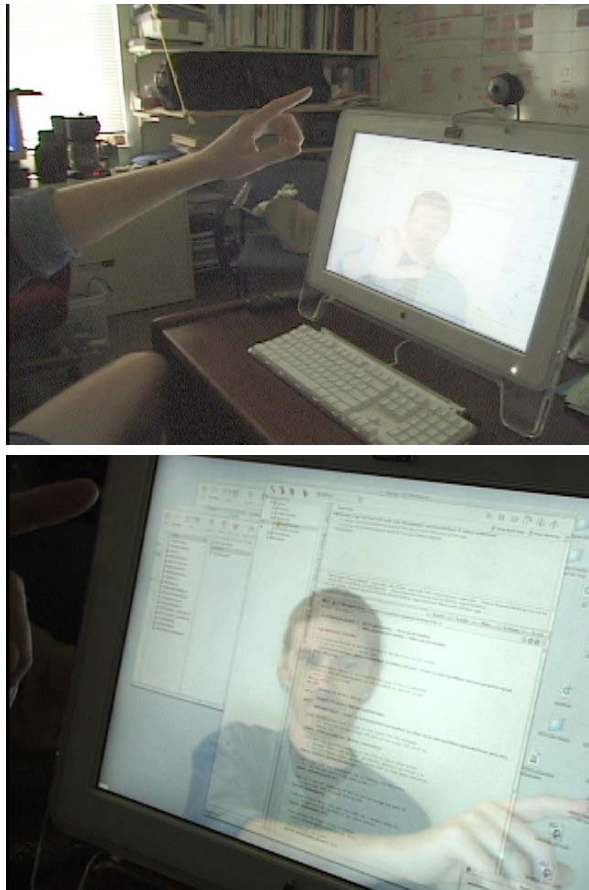


Figure 1: Facetop physical setup, with iBot video camera

Using image analysis techniques we track the user’s fingertip and optionally drive the mouse from this tracker. Figure 2 shows this finger tracking during Web browsing. These screens also show that the user can varying the levels of transparency, giving different emphasis to the desktop contents relative to the user image.

A fully opaque Facetop (all user, no desktop) is purely a communication tool, and is especially useful in the two-head version (next section) for allowing collaborators to speak face-to-face about a task without application window clutter. A fully

transparent Facetop is the same as a normal desktop, showing no user video image. Most uses for the Facetop will involve a semi-transparent video setting, giving a mix of user image and desktop content on the screen.

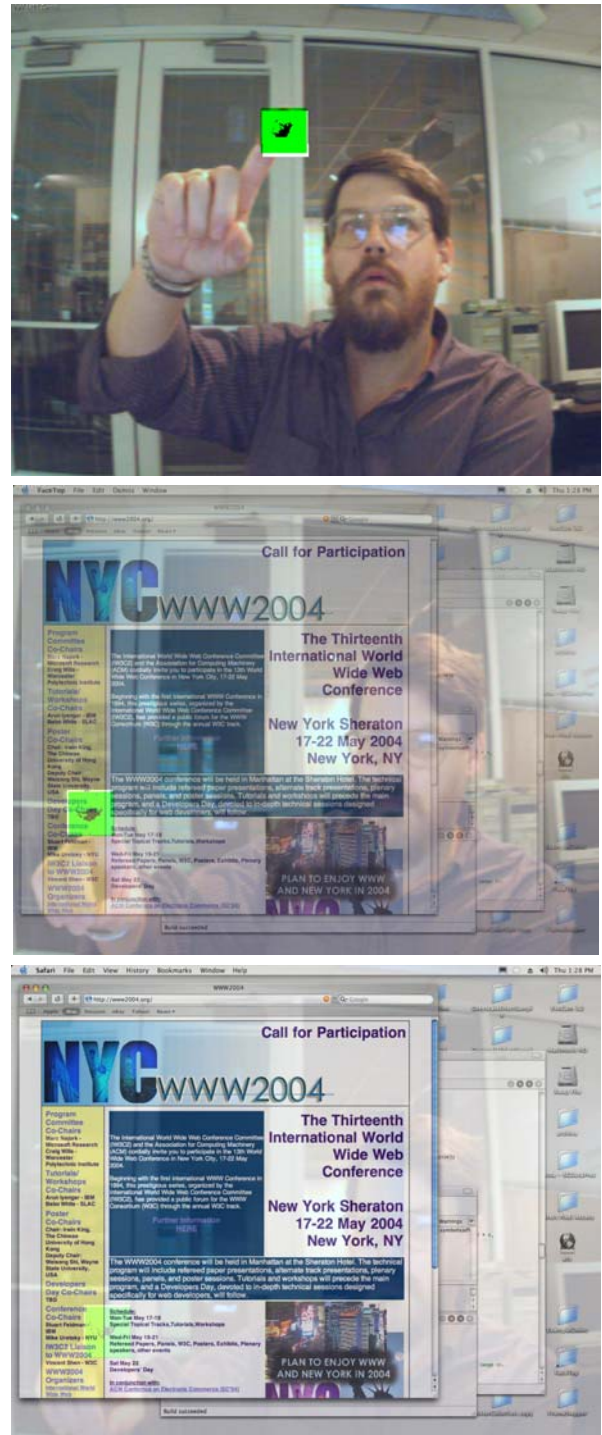


Figure 2: Varying user transparency, opaque to clear

Mouse clicking and Auto Video

When finger tracking is turned on, Facetop users can drive the mouse via direct pointing. This includes clicking the mouse as well as moving it. We have a click mechanism that involves a pinching

motion that partially occludes the fingertip. The pinch initiates a timer. If the tip re-appears in ½ second, it is a single-click; if it reappears between ½ and 1 second, it is a double click; if more than 1 second goes by, the tracker decides the user has removed the finger from camera view and goes back to home state.

Another user-selectable feature is the appearance of the video itself. In auto mode, the user video stream is not composited into the desktop until the user finger is raised into camera view. At that point, the Facetop view begins, with user video and desktop composited semi-transparently. When the user drops the hand from camera view, the Facetop reverts to full desktop only. This mode helps control video clutter, making video visible only when needed for pointing. It is effective for presentations, where the user not needed constantly for communication. Auto mode requires finger tracking, but does not necessarily require mouse movement as the finger tracked. These two are independently selectable.

Two-head Collaborative Facetop

Though the discussion so far has been in the context of a single-user PC interface, another useful domain of application for the Facetop is in collaborative systems – specifically for supporting synchronous *paired* activities like distributed pair programming discussed earlier.

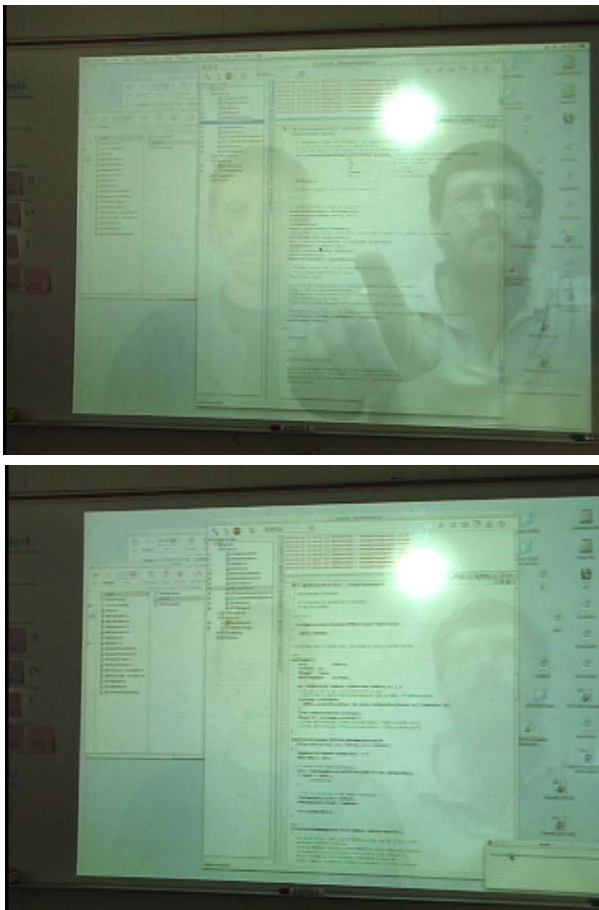


Figure 3: Pointing, varying transparency in 2-head Facetop

The Facetop, via its video capabilities, provides potential solutions to the three problems mentioned earlier for dPP (lack of facial expressions, pointing, and whiteboard). In Facetop the video image

is large, and frame rates run from 18 to 30 fps (depending on host network speeds), showing facial details and fine motor movements of the fingers and lips. The video image is also tightly and seamlessly integrated with the shared workspace via transparency, thereby eliminating the “dual” nature of video teleconferencing solutions. Users do not have to switch their attention from desktop, to video, back to desktop, etc.

For the dual-user Facetop, we have built a setup that has both video streams (each collaborator) composited with a shared desktop; figure 3 shows this version in a projected environment. Each user sits slightly to the right so that the two heads are on different sides of the frame when the two video streams are composited. We also sit each user against a neutral background to control the possible added visual confusion. Collaborating users continue, as before, to communicate audibly while using the Facetop via an Internet chat tool like Yahoo Messenger.

The primary advantage the Facetop gives over other approaches is the close coupling of communications capabilities with examination of the content. Each user can see where the other points in the shared workspace; they can also use the Facetop as a direct video conferencing tool (by varying the transparency level to fade the desktop image) without changing applications or interrupting the work activities. We have key presses and voice commands to quickly vary these levels.

As mentioned previously, finger tracking and mousing is a user selectable mode. For collaborative use, the Facetop is effective for pointing and communication even without finger tracking.

System structure and performance

Our implementation is simple, and potentially ubiquitous due to its modest equipment needs. Facetop uses a \$100 Sony iBot camera, and runs with excellent efficiency on an Apple Powerbook, even when processing 30 video frames a second. No supplemental electronics are needed for wearing on the hand or head for tracking or gesture detection. Facetop is minimally invasive on the user’s normal mode of computer use.

The advantages of a Macintosh implementation are that the desktop is rendered in OpenGL, making its image and contents not private data structures of the OS, but rather available to all applications for manipulation or enhancement. We also use dual-processor platforms, so that one processor can handle tracking issues and other Facetop-specific loads, while leaving a processor free to support the collaborative work, such as pair programming. Video processing is handled mostly on the graphics card.

Some of our experiments have been run between two Power Mac’s connected via peer-to-peer gigabit network. In this configuration, we get a full 30 fps video in each direction. This is possible due to the high network speeds, to passing only the 640 x 480 camera image. Image scaling to screen size is handled locally on each machine after the 2 video signals and the desktop are composited into one image. Similar experiments have been done on the normal switched 100-mb Internet in our lab. In this mode we get performance of 18 frames per second. This is acceptable for pointing, but not for more complex communications such as signing or lip reading (for dPP by people with audio impairments).

Initial User Evaluations

Controlled user evaluations are ongoing, but we have some initial usability results to report. We have had 15 users try the basic

Facetop to determine if live background video is a viable, usable concept as an interface for manipulating the PC environment. We set up in a room with white walls so that there would not be a busy background to add visual clutter to the screen image.

As might be expected, arm fatigue is a problem for continuous use of the fingertip-based mouse feature. For browsing the Web, or discussing code, this is not a major issue, as much time is spent reading vs. actually manipulating the screen. Users drop their arm during these quiescent periods, and then raise it to point when ready to navigate more. The video on-screen gives the visual cues needed for positioning the mouse pointer directly where needed.

Another potential problem is visual clutter. Most users adapted quickly and comfortably to the moving image in the background; different users set transparency at different levels, and there did not seem to be a preferred level of mixing of desktop with user-image other than to say that both were visible. The human eye/brain is able to pay attention to (or ignore) the face or the desktop respectively, depending on the cognitive task – depending on whether the user wants to read the screen contents or to communicate (in the two-head version). Users were queried specifically as to visual clutter or confusion. A few objected, but most found the adjustability of transparency fine-grained enough to get to a level where they were not distracted or hindered.

Dual-head trials

We created a simple networked tic-tac-toe game for usability trials of the dual head version and had 11 pairs of users try it. The users were a class of 8-grade students who came to the department for research demonstrations. Five of the users took less than 5 minutes to become facile with the interface, learning to move and click the mouse well enough to Web browse. All users were able to successfully play the game (which involves clicking on GUI buttons) in the 30-minute time frame of the trials.

We had five of the programming pairs involved in past dPP experiments (with audio and shared desktop only) try the Facetop environment for small pair programming “shakedown” tasks. Since all had tried the earlier environments, the trials were designed to see if the “video made large” features in Facetop overcame the lack of pointing ability and lack of facial expressions reported by these teams before (the lack of whiteboard they reported is still being investigated with other features). All teams were quite comfortable using the Facetop, and did not consider visual complexity or clutter an issue. We suspect this is due to concentration on programming focusing the attention on the various text windows of the desktop. All dPP teams were able to complete small programs with no problems.

No teams chose to completely fade out the video and use audio only. All teams left the user images visible to some extent and did use the video to point to code being discussed. In post-trial interviews, the overall impression was that Facetop was an interesting improvement over the audio-only dPP environment used before. Each team was asked, “If you were to do a longer dPP development, would you prefer to use Facetop or the original audio-only environment?” All expressed a preference for Facetop. These simple usability trials do not reveal if the preference for Facetop was emotional or qualitative only, or if the added video and sense of presence increases programmer effectiveness. We find these early usability trials compelling enough, though, to start larger, controlled

experiments to see if Facetop can have an impact on quantitative aspects, such as design quality or error counts.

Acknowledgements This work was partially supported by a grant from the U.S. Environmental Protection Agency, # R82-795901-3.

References

- [1] Beck, K., *Extreme Programming Explained*, Addison-Wesley, 2000.
- [2] Stotts, D., J. Smith, and K. Gyllstrom, “Single- and Dual-User Web Browsing in the Transparent Video Facetop,” TR04-005, Computer Science Dept., UNC, Dec. 2003, <http://rockfish.cs.cs.unc.edu/pubs/TR04-005.pdf>
- [3] A. Cockburn and L. Williams, “The Costs and Benefits of Pair Programming,” *eXtreme Programming and Flexible Processes in Software Engineering -- XP2000*, Cagliari, Sardinia, Italy, 2000.
- [4] H. Ishii, M. Kobayashi, and J. Grudin, “Integration of inter-personal space and shared workspace: ClearBoard design and experiments,” *Proc. of ACM Conf. on Computer Supported Cooperative Work*, Toronto, 1992, pp. 33-42.
- [5] R. S. Fish, R. E. Kraut, and B. L. Chalfonte, “The VideoWindow System in Informal Communications,” *Proc. of ACM Conf. on Computer Supported Cooperative Work*, Los Angeles, 1990, pp. 1-11.
- [6] P. Baheti, L. Williams, E. Gehringer, and D. Stotts, “Exploring the Efficacy of Distributed Pair Programming,” XP Universe 2002, Chicago, August 4-7, 2002; Lecture Notes in Computer Science 2418 (Springer), pp. 208-220.
- [7] P. Baheti, L. Williams, E. Gehringer, D. Stotts, “Exploring Pair Programming in Distributed Object-Oriented Team Projects,” Educator’s Workshop, OOPSLA 2002, Seattle, Nov. 4-8, 2002, accepted to appear.
- [8] Eric A. Bier, Ken Fishkin, Ken Pier, Maureen C. Stone, “A Taxonomy of See-Through Tools: The Video, Xerox PARC, Proc. of CHI ’95, <http://www.acm.org/sigchi/chi95/Electronic/documnts/videos/eab1bdy.htm>
- [10] Beverly L. Harrison, Hiroshi Ishii, Kim J. Vicente, and William A. S. Buxton, “Transparent Layered User Interfaces: An Evaluation of a Display Design to Enhance Focused and Divided Attention,” Proc. of CHI ’95, http://www.acm.org/sigchi/chi95/Electronic/documnts/papers/blh_bdy.htm
- [11] Stotts, D., L. Williams, et al., “Virtual Teaming: Experiments and Experiences with Distributed Pair Programming,” XP Universe 2003, New Orleans, Aug. 10-13, 2003, LNCS 2753 (Springer), pp. 129-141.
- [12] Stotts, D., J. McC. Smith, and D. Jen, “The Vis-a-Vis Transparent Video FaceTop,” UIST ’03, Vancouver, Nov. 3-6, 2004, pp. 57-58.
- [13] Nosek, J.T., “The Case for Collaborative Programming,” *Communications of the ACM*, March 1998, pp. 105-108.
- [14] Olson, G.M., and J.S. Olson, “Distance Matters,” *Human-Computer Interaction*, vol. 15, 2000, pp. 139-179.
- [15] Roussel, N., The videoSpace Toolkit, Web information <http://www.lri.fr/~roussel/projects/videoSpace/>