# DiamondSpin :
## An Extensible Toolkit for Around-the-Table Interaction

Source : Conference on Human Factors in Computing Systems (CHI '04)

Authors : Chia Shen Mitsubishi Electric Research Labs, Cambridge, MA
Frédéric D. Vernier University of Paris, Orsay, France
Clifton Forlines Mitsubishi Electric Research Labs, Cambridge, MA
Meredith Ringel Stanford University, Stanford, CA

Publisher : ACM

# INDEX

- Abstract
- Introduction
- Motivations and Background
- The DiamondSpin Toolkit
- DiamondSpin Applications
- Discussion of User Experience
- Related Work
- Conclusion

# Abstract

- DiamondSpin
  - Toolkit for the efficient prototyping of experimentation with multi-person, concurrent interfaces for interactive shared displays.
- In this paper,
  - Identify the fundamental functionality that tabletop user interfaces should embody.
  - Present the DiamondSpin toolkit's architecture, API, and applications.
    - novel real-time polar to Cartesian transformation engine.
    - arbitrary document positioning and orientation.
    - polygonal tabletop layout can easily be constructed.
    - multiple work areas, multi-user operations.
  - Discuss insights on tabletop interaction issues.

# Introduction

- Goal in creating the DiamondSpin toolkit is twofold:
  - it is meant to allow us to further explore fundamental issues regarding the design of tabletop interfaces.
  - it is also intended as a toolkit to enable others to quickly build multiuser tabletop applications.
- Tables are a familiar piece of furniture and physical setting for people to meet, chat, look over documents, and carry out tasks that require face-to-face collaboration.
- Digital documents are commonly used only on single user desktop computers and handheld portable devices, due to a lack of support for face-to-face around-the-table applications.

# Motivations and Background

- Tabletop user interfaces should preserve many of the familiar and useful properties a physical tabletop affords and allow the natural interaction that people usually carry out on physical tables.
  - reorienting objects, passing documents, spreading & piling documents.
- It is very difficult for a typical programmer or graduate student to create even the most basic interface components that embody some of the fundamental affordances of common tabletop interactions.

# Motivations and Background (cont.)

- Fundamental tabletop UI functionality – PDH

| Requirement | Examples |
|---|---|
| Visual document management | Arbitrary directional orientation of document placement.<br>Document layout. |
| Document control and interaction | Visible controls allowing direct document manipulation.<br>Direct passing of individual objects as well as groups of objects among the users around the table. |
| Manipulation | Bare-hand, stylus and keyboard all should be supported. |
| Rotational UI | Managing user interface components that are either rotatable or rotation sensitive (UI components that contain text whose readability would be affected if automatically rotated away from the user, such as a menu bar). |
| Digital tabletop layout | Creating and managing private and public spaces.<br>Creating and managing multiple virtual tabletops. |
| Multi-user support | Multiple menu bars.<br>Concurrent multi-user interaction techniques. |

# Motivations and Background (cont.)

- Conventional UI software is based on rectangular displays for single-user usage scenarios.
- A tabletop user interface is for multiple users interacting concurrently and may not necessarily have rectangular views.
- Thus, many of the display algorithms and image composition algorithms for optimized rendering must be modified, and new architectural support must be designed.

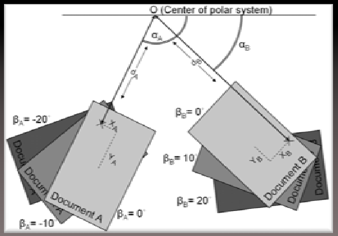
# The DiamondSpin Toolkit

- When multiple people gather around a table, there is no single directional viewing angle or orientation that is ideal for everyone present.
- At the heart of DiamondSpin is a polar-coordinate system that enables continuous individual document orientation among multiple people with arbitrary viewing angles, as well as the rotation of the entire table surface.
- DiamondSpin Architecture
  - Two engines for
    - real-time transformation of polar coordinates into a standard transformation matrix for graphics context and input events.
    - handling multi-layer multiple depth display functions, and coordinating multiple threads and tabletop views.

# The DiamondSpin Toolkit (cont.)

- Transformation Engine
  - In DiamondSpin, every element in the display are described in terms of a polar distance and a polar orientation.
  - Two key concepts:
    - Translation of the origin of the conventional Cartesian display (usually at the top left or bottom left corner) to the center of the tabletop display,
    - Three degrees of freedom (3 DOF) d, α, and β for each element on the tabletop.

- d: the distance to the table center.
- α: the orientation angle with respect to table center.
- β: the rotation angle of the element with respect to its own rotation point.

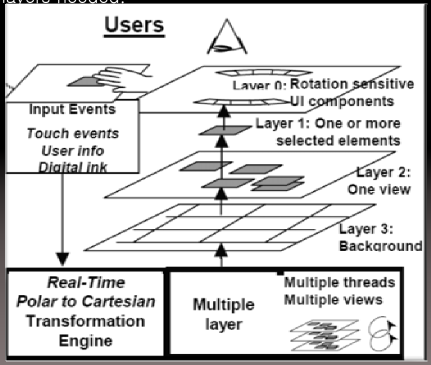


# The DiamondSpin Toolkit (cont.)

- To compute the relative position and orientation of each element, the translation from a position (d, α) into a transformation matrix is carried out.
- Given d, α, and β, our transformation engine carries out the following affine transformation for each document on the table.
  - $T^t_{(width/2,\ height/2)} \circ S^t_{(-1,\ +1)} \circ R^t_{(0,\ 0,\ \alpha+\varphi)} \circ T^e_{(d,\ 0)} \circ R^e_{(0,\ 0,\ \pi/2+\beta)} \circ T^e_{(ElemWidth/2,\ ElemHeight/2)} \circ S_{(-1,\ -1)}$
- This engine employ the same matrix to compute the inverse transformation that translates the input points from the Cartesian screen space into the tabletop polar coordinate element space, in order to determine the target document for an input event.
- The three degrees of freedom, (d, α, β), enable DiamondSpin to support tabletop-specific document handling interactions.
  - a document passing metaphor

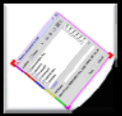
# The DiamondSpin Toolkit (cont.)

- Multi-Layer Multi-Thread Display Management
  - This architecture allows DiamondSpin to selectively refresh part of a tabletop display, thus implementing efficient user interaction schemes and allows an application to use only the layers needed.


Users
Input Events
Touch events
User info
Digital ink
Layer 0: Rotation sensitive UI components
Layer 1: One or more selected elements
Layer 2: One view
Layer 3: Background
Real-Time Polar to Cartesian Transformation Engine
Multiple layer
Multiple threads
Multiple views

# The DiamondSpin Toolkit (cont.)

- Discussion of Implementation
  - DiamondSpin is implemented in pure Java 2D with JAI and JMF.
  - Java is platform-independent: a portable and extensible toolkit.
  - mouse input events for conventional input devices.
  - explicit support to run on DiamondTouch, a multi-user, multi-touch input surface with unique user identification capabilities.
  - mechanism to transform Java Factory Components into DiamondSpin's polar coordinate system.
    - The DiamondSpin framework can display Java Swing components at an arbitrary orientation.
    - Rotation of GUI elements in DiamondSpin is achieved through replacement of location functions with polar coordinate computation as well as subclassing of existing Swing Java classes.
  - concurrent user input events are handled by multiple threads, one per user with a unique thread name.

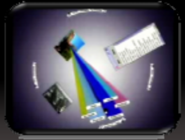


# The DiamondSpin Toolkit (cont.)

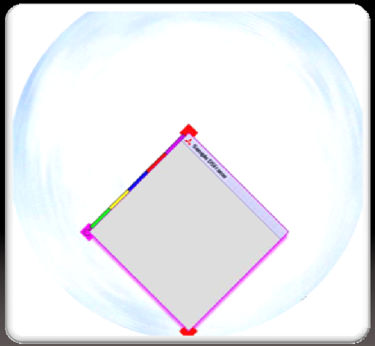
- DiamondSpin API
  - DSContainer (main class)
    - methods for input event handling and repaint/refresh of the display.
    - methods to handle document orientation by allowing programmers to specify the angle at which an element should be rotated.
    - methods to handle "magnetization" which allows the developer to specify a global angle to which all documents should align themselves.
    - DSContainer.startRotateTable(angle,userID); // rotate the entire display
  - DSView
    - create and manage multiple views within the same application.
    - create multiple personal and shared work areas within the same virtual tabletop display.
    - view is an object instantiated by DSContainer.
      - receives input events and executes repaint orders from the DSContainer.
      - provides methods to set a different background image, pan, scroll or rotate the view, and open a contextually appropriate popup menu on an element or on the background.

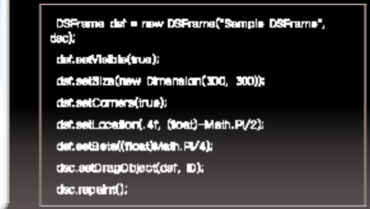
# The DiamondSpin Toolkit (cont.)

- DSFrame, DSPanel, and DSWindow
  - JFrame, JPanel, and JWindow.
  - replaces location functions with polar coordinate ones and adds angle of orientation and zoom factor functionalities.
  - provide optional attributes such as colored shadow generation and finger-size corner handles (visual feedback to indicate that a particular document is actively being selected and used) for resizing or re-orientation.
- DiamondSpin provides two types of menus:
  - DSMenuBar
  - DSPopupMenu
- DSStroke
  - digital ink
- DSKeyboardPanel
  - popup keyboard



# The DiamondSpin Toolkit (cont.)

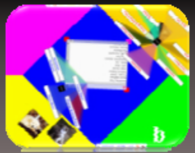


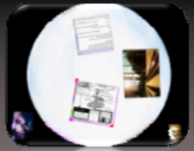


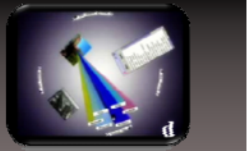
```
DSFrame def = new DSFrame("Sample DSFrame",
dsc);
def.setVisible(true);
def.setSize(new Dimension(300, 300));
def.setCorners(true);
def.setLocation(.4f, (float)-Math.PI/2);
def.setBeta((float)Math.PI/4);
dsc.setDragObject(def, ID);
dsc.repaint();
```

# DiamondSpin Applications

- Table for N
  - This application is used for a small number of people (two to four) sitting around a table collaboratively creating, sketching, annotating, manipulating, and browsing various types of documents, including text, html, images and video clips.
    - (a) a rectangular tabletop
    - (b) a tabletop with 1 shared center, 4 personal work areas, and 2 popup menus
    - (c) a rotatable circular tabletop
    - (d) a rectangular tabletop with continuous document orientation.





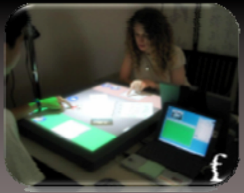


## DiamondSpin Applications (cont.)

- An Opportunistic Browsing Coffee Table
  - The environment for opportunistic browsing is a coffee table constructed with DiamondTouch hardware and DiamondSpin software.
  - A continuous stream of information items moves slowly along a circular path across the table surface.
  - If an information item is interesting to a user, he can move it into the center whereupon more detail becomes available for closer examination.
  - The implementation of the coffee table only uses one DSView.
  - The DiamondSpin application only has to rotate the view through the DSContainer's setAngle method and add new elements in the view when the table has rotated for a specified angle.

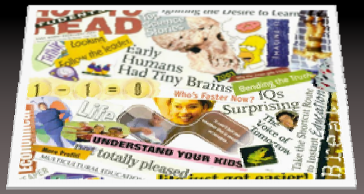
## DiamondSpin Applications (cont.)

- UbiTable
  - It is a multi-device application built using the DiamondSpin toolkit.
  - People can walk up to a UbiTable with their laptops and/or USB devices such as cameras.
  - Users can collaboratively layout, annotate, and mark up content from their own devices on the UbiTable. They can also easily exchange content among laptops and USB devices with others sitting around the table.
  - On UbiTable, each user is provided with a private work area, while the center of the table is a shared area.
  - Thus UbiTable extends DSView to include sub-views.



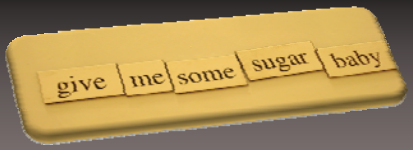
## DiamondSpin Applications (cont.)

- A Tabletop Collage and Webpage Builder
  - It is a direct-manipulation tabletop design and layout application.
  - Users combine images and text at arbitrary sizes and orientations, and export this work as a web page that uses the collage as an image map linking from each of the collage components to a full-sized version of the corresponding source documents.
  - The collage is derived from the DSElementGroup class in DiamondSpin, which provides facilities for piling several objects into a single unit.



## DiamondSpin Applications (cont.)

- PoetryTable
  - It is an educational game, inspired by the popularity of “magnetic poetry”.
  - The game allows up to four simultaneous users to combine a set of English or Japanese word tiles to create poetry.
  - The word tiles (PoetryMagnet) are automatically rotated to face each side of a rectangular tabletop.
  - Popup menus supported by DiamondSpin give users the option to make duplicates of popular word tiles, to add a suffix or prefix to a particular word, and to save a screenshot of the game in order to preserve their poems. (PoetryGenericView)



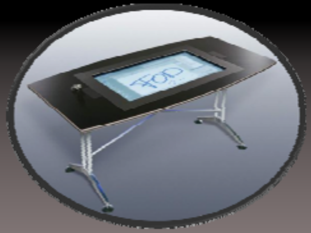
## Discussion of User Experience

- Through the experience of using DiamondSpin, multi-user shared tabletops, many interesting interface design and interaction technique issues have become apparent.
  - Conflicts, even though often unintentional ones, among users are an issue that DiamondSpin has exposed.
  - In a traditional single-user interface, a document is selected when its title bar and frame are highlighted. This metaphor may not translate directly into a tabletop UI.
  - When building applications that use DSPopupMenus to present contextually relevant choices, it is not clear what the best way is to invoke these menus. This has implications for designing command invocation methods in general on a touch-sensitive surface.

## Related Work

- The Living Memory (LiMe) project

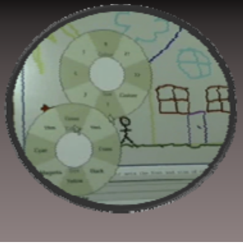
- The InteracTable and the ConnecTable in the i-Land project



## Related Work (cont.)

- DigitalDesk

- MID (Multiple Input Devices) Java Package and SDGToolkit



## Conclusion

- DiamondSpin Toolkit's architecture and API.
- A novel real-time polar to Cartesian transformation engine that enables around-the-table interactions with arbitrary document positioning and orientation on a tabletop surface
- DiamondSpin has proven to be a versatile toolkit to study, build, and experiment with interactive tabletop applications.
- We will further develop conceptual models and UI components that can be incorporated into the toolkit.
- The affordances of a digital tabletop are a new territory.