CHALMERS

Dual-Mode Infrared Tracking of Tangible Tabletops

Master of Science Thesis in Secure and Dependable Computer Systems

Seyed Ali Alavi

Chalmers University of Technology
University of Gothenburg
Department of Computer Science and Engineering
Göteborg, Sweden, October 2012
Report No. 2013:005
ISSN: 1651-4769

Dual-Mode Infrared Tracking of Tangible Tabletops

Seyed Ali Alavi

Examiner: Morten Fjeld

Chalmers University of Technology
University of Gothenburg
Department of Computer Science and Engineering
SE-412 96 Göteborg
Sweden
Telephone + 46 (0)31-772 1000

Cover: Kuggen, located in Lindholmen, Göteborg, where this project was accomplished.

Department of Computer Science and Engineering
Göteborg, Sweden October 2012

Dual-Mode Infrared Tracking of Tangible Tabletops

Supervisor: Prof. Morten Fjeld

March 19, 2013

Abstract

On tangible tabletops, Tangible User Interfaces (TUIs) can signalize their identity, position, orientation, and state by active infrared light. This provides rich interaction capabilities in complex, dynamic scenarios. If TUIs have to transfer additional high-resolution information, many bits are required for each update. This has a negative impact on the overall update rate of the system. In this thesis, we present a new solution for providing TUIs with high number of states. Prototypical TUI concepts such as slider, ruler, and dials further motivate the benefit of high-resolution tracking. We depart from a device tracking overview and then show how tangible devices for tabletops typically use infrared (IR) emitters and a camera to send information about their position, orientation, and state. Since transferring many additional information bits via a normal camera-based tabletop system is not feasible anymore, we introduce next a new system setup that still offers a sufficiently high update rate for a smooth interaction. The new method can be realized as a tabletop system using a low-cost camera detecting position, combined with a low-cost infrared receiver detecting the state of each device. Since both kinds of sensors are used simultaneously we call the method "dual mode." This method combines a camera-based tracking with the possibility to transfer a significantly high amount of states for each device.

Contents

1 Acknowledgment

The work presented in this document could not be done without the help of many great people.

Professor Morten Fjeld provided all kinds of support, and helped developing the context of this thesis at t2iLab in Chalmers University of Technology.

PD. Dr. Andreas Kunz, whom his constant review of this text and offering his valuable feedbacks was only one of his pivotal helps in writing this thesis.

Tommaso Piazza, Anders Rudback, and all other students in t2iLab, who offered me their help whenever I asked for.

I am thankful to all of them.

2 Introduction

Tabletop computing systems are one of the mainstream form factors of ubiquitous computing. The introduction of Microsoft Surface in 2008 was a big step toward making this form of tabletops a popular way of personal computing.

A tabletop system is a computing system with a form factor of a table, consisting of a computer, a display device (usually a back projection screen or an LCD), which provides the users an interactive surface for entering their inputs. This interactive surface is the same surface as the display surface.

In the mainstream of tabletops, the user interactions are by touch, or by using tagged objects. This requires a tracking mechanism. The computer system needs to detect touches and objects to interpret user inputs.

This master thesis presents the design and implementation of a novel algorithm for tracking Tangible User Interfaces (TUIs) on tabletops. On tangible tabletops, TUIs can signalize their identity, position, orientation, and state by actively emitting infrared light, hence having the ability to transform a few information. This provides rich interaction capabilities in complex, dynamic scenarios. However, if TUIs have to transfer additional high-resolution information, much more bits are required for each update. This has a negative impact on the overall update rate of the system, since one camera acquisition frame is required for each bit. In this thesis, after presenting some background and applications of TUIs on tabletops, an overview of device tracking on tabletops, particularly using infrared (IR) emitters and a camera, is provided. Furthermore, the drawbacks of current technologies are discussed, and it is shown that sending many additional information bits via a normal camera-based tabletop system is not very efficient, since it results in an unacceptable high latency. Later, the theory behind the new system setup that offers a sufficiently high update rate for a smooth interaction even for multi-bit TUIs is discussed. The new working principle is again realized as a tabletop system. Since two sensors for different purposes are used, the method is called "dual mode." It combines a camera-based tracking with the possibility to transfer a significantly higher number of states for each device. Finally, the thesis presents implementation challenges and details of software and hardware, and provides some recommendations for future work.

This thesis lead to some publication in international conferences, and one journal article is to be submitted shortly. The published papers are presented in the appendix.

3 Background

3.1 Related work

In order to enable intuitive interaction with the tabletop content, devices other than the mouse and keyboard must be used. There is a class of devices that are easily identifiable by their inherent function known as 'physical icons' or phicons [4]. In this case, each device usually has a static association so that the tabletop system is able to detect its identifier (ID) in addition to its position. Once the device's ID is known to the system, the underlying functionality is also defined since the association cannot be changed. However, other tabletop systems have a dynamic association that allows for simpler detection algorithms. In the latter case, the devices have a more general character, and so the intuitiveness is only guaranteed by the displayed content, i.e., the graphical user interface (GUI). The dynamic association is user-triggered and follows predefined steps. These steps may require some learning on the part of the user.

In [7]it was stated, that tabletop systems must be able to detect the positions of interaction devices and, in the case of phicons or other specialized input devices, also their ID. While the position of a device is important for the interaction in a global context, the ID is relevant for integrating a device's specialized functionality into a specific application.

Although tracking has been well researched in the field of virtual reality, it is still a delicate task even on a 2D tabletop surface. More degrees of freedom (DOF) than given by planar interaction become relevant. For instance, the z-coordinate may be used to distinguish between writing and pointing in pen-based interaction. Additionally, the tracking and detection system's latency should be below the user's perceptual threshold, otherwise user irritation may occur.

An even more critical task for the tracking and identification system is distinguishing between objects meant for interaction, such as a finger, and objects not meant for interaction, such as coffee mugs or the side of a user's hand. During normal operation of a tabletop system, various objects may be placed on the surface which are not meant for interaction, but could interfere with the system, e.g., by shadowing effects. Unlike a mouse, which is a relative pointing device (i.e. the travelling distance and orientation are detected), all tracking systems for tabletop systems allow absolute pointing, i.e., the object is detected at precisely the place where the user puts the device.

Active Desk [2]was one of the first systems that allowed the usage of multiple TUIs on a back-projected tabletop using an electromagnetic tracking system. Two years later, Ullmer et al. [10] introduced the metaDESK, which used an infrared vision-based tracking to detect specialized devices on the tabletop together with an electromagnetic tracking system to track devices above the table's surface such as a magnifier lens. One of the first systems that used only vision-based tracking for localization and identification of objects

was reacTable [6]. All specialized interaction devices were applied with so-called fiducials, which is a unique b/w-pattern that can be simply recognized by the camera underneath the tabletop. Thus it was possible to detect position, orientation, and ID of each TUI. Such a visual code was also used by Wilson's PlayAnywhere [11] to detect individualized devices on a front-projection system. For a brainstorming application, InfrActables [3] provided users with active TUIs. In that system, TUIs have form factors, such as pen, handle, brick, ruler, and color tool, which may enable a more natural interaction style. In this system, once the devices are triggered by an infrared signal, they are able to transmit state and ID on a different IR-wavelength. A camera under the table captures the response, and the results of the time-dependent blob-detection (five frames) give information about position, ID, and state. In this protocol, a device with 2 bits for the ID and 3 bits for the state would require 5 acquisition frames as shown in figure 3.1.

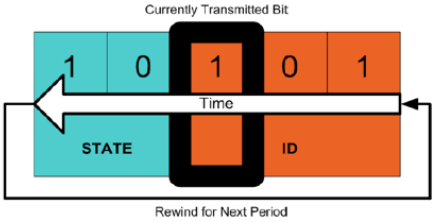


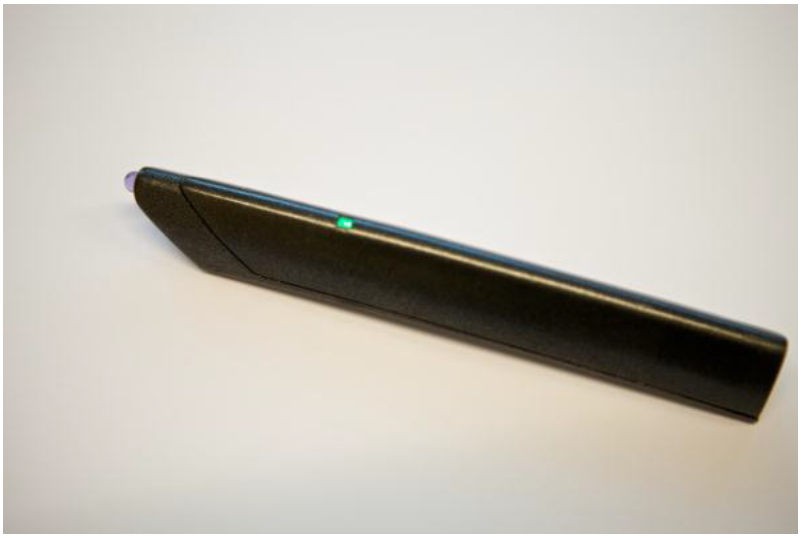
Figure 3.1: Simplified protocol

However, when realizing a system using a low-cost camera with, for example, a frame rate of 60 Hz, the overall device-state refresh rate is 12 Hz, even if only one single device is use. This is at the lower limit for human tabletop interaction.

InfActables was the first tabletop system that used TUIs with multiple states, since each of these TUIs has a few state-triggering widgets, such as buttons on top of the bricks or a micro switch under the pen's tip. Thereby, users can control the TUI's state while positioning it on the surface. However, tracking of TUIs is so far limited to the spatial and temporal resolution of the camera's CCD chip. While this is sufficient for detecting the position of TUIs, it becomes problematic when detecting the ID of a device. Since fiducials need a certain size for displaying a recognizable unique pattern, they do not allow realizing small objects because they could not be resolved by the camera anymore. On the other hand, the temporal approach to signalize ID and state information of a TUI relies on the camera's frame rate, which should be very high to avoid disturbing latencies. These problems are addressed in more recent tabletops, which also inspired this thesis: QualiTrack[8], MightyTrace[9], and SmartFiducial[5].

3.1.1 QualiTrack

QualiTrack is a tabletop system that works with infrared signals. It tracks the IR devices using a high speed infrared camera. The image is back-projected on the screen using a projector located under the table. QualiTrack offers a variety of TUIs, including a simple stylus used for pointing, drawing, and writing, a frame, used to zoom in its surrounded area, and a color picker (see figure).



(a) Stylus

(b) Frame

(c) Color picker

Figure 3.2: Tangible devices for QualiTrack

Each device has an infrared receiver, and one or three infrared emitters, depend on whether the device orientation matters or not. Moreover, each devices is controlled with an embedded microcontroller.

The system also has a central controller unit, called Sync Unit, used for synchronizing the camera and the devices. The synchronization is made by triggering a block of infrared LEDs located under the table.

TUIs then emit a fixed-length (10 bits) binary code, bit per bit, upon every synchronization signal. This bit code consists of an unique 8-bit device ID code, and a 2-bit state information, indicating the state of the push buttons on the device.

The camera captures a frame at every synchronization signal. After capturing as many frames as the bit code's length (10), the host computer can determine the position, the ID, and the state of the devices. Also, the orientation of the devices with three LEDs can be determined by using a simple image analysis.

Since the camera speed determines the update rate of the system, a high speed camera, namely QualiSys MCU1000, is used. The camera is capable of capturing up to 1000 frames per second. Although, since there is a trade off between the frame rate and the image resolution (the higher the frame rate, the lower the resolution), the camera is operated at 250 frames per second, with a resolution of 658x496 pixels.

This leads to an update rate of 25 frames per second. Although this update rate is enough for reading the ID and state of the devices, it is rather slow for tracking the position of the devices. Observing that it is not necessary to receive all eight ID bits to identify each device, device positions are updated as soon as the device is identified. For example, the system can distinguish between 01010101 and 10101010 by reading every two consecutive bit of each code, improving the update frequency of the system by a factor of 4.5, i.e. 125 frames per second. Moreover, it is important to note that the devices doesn't send any infrared signal when transferring a 0 bit, thus the presence of 1 bits in a bit code helps to track the devices more efficiently.

Finally, the selection of device IDs plays an important role in the update rate of QualiTrack. Also, the update rate of devices' position and state is directly correlated to the camera's frame rate. [8]

3.1.2 MightyTrace

In MightyTrace, a matrix of IR sensors is used to detect position, ID, and state of TUIs in a time-multiplexed approach. Each device is assigned a specific number of time frames during which its LED is turned on. Typically, one (two if the device's rotation is to be communicated) frame is used for the ID. Thus, each device can be detected unambiguously. For sending states, there are even more time frames for each device [9]. However, the main restriction with such a time-multiplexed approach is that the system update rate is limited not only by the amount of devices (and their types) on the surface, but also by the number of states each device may have. Furthermore, Hofer et. al.'s solution [9] was realized by using an array of IR-sensors instead of a camera, which had the advantage of being integratable in an LC-screen. However, the drawback is that this solution is very expensive and not applicable to back-projection systems due to shadow-

casting of the sensors. Still, the system introduced the convincing idea of using fast semiconductors for detecting information that is emitted from the devices. This allows using an individual time interval for each device (serial interrogation of information), while still having a sufficiently high update rate (See figure 3.3).

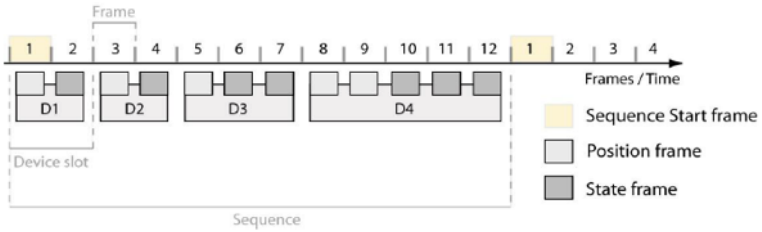


Figure 3.3: Data communication in MightyTrace

3.1.3 SmartFiducial

SmartFiducial is "a wireless tangible object that facilitates additional modes of expressivity for vision-based tabletop surfaces" [5]. They are used together with an application in an interactive musical setting.

The object tracking in 2D space (X, Y and rotation) is vision-based, realized by a custom version of **libfidtrack** engine (developed for the reacTIVision system). The engine is implemented in the open-source vision tracking software CCV (Community Common Vision).

SmartFiducial also detects its Z-depth using in infrared proximity sensor. Moreover, it can detect pressure-based gestural input using two pressure sensitive buttons (See figure [5]). The pressure and Z-Depth data are delivered to the host computer using an XBee wireless transmission module.

Since SmartFiducial uses two different mediums for transferring different type of information (position and orientation visually, and Z-depth and pressure data via XBee), the communication system is rather complicated. Moreover, there is always a risk of unintended interference with RF communication. Finally, depending on the settings, the wireless communication may be subjected to specific regulations (usage of some radio frequencies is prohibited in hospitals, military bases, ...).

3.2 Usage scenario

Since designing an interactive system relies on different usage scenarios, and on how users interact with such a system, performing a user study could be an invaluable asset. This section presents such a user study, performed on InfrActables setup.

On this system, four different interactive devices were realized: pen, ruler, color tool, and notepad. See figure 3.5 .

For the user study, the principle presented in [1] is followed, which proposes to evaluate a collaborative setup by measuring the performance (completion time), and other

Figure 3.4: SmartFiducial prototype [5]

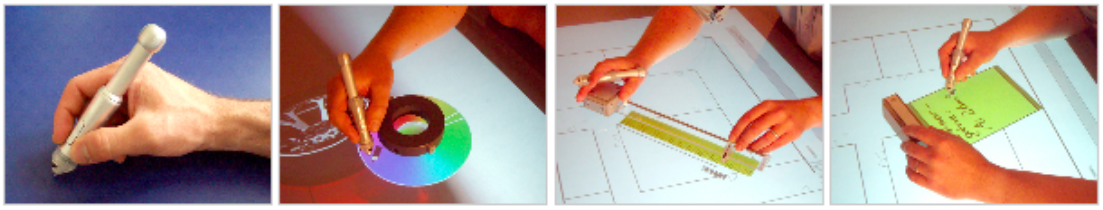


Figure 3.5: Interaction devices realized for InfrActables

subjective measures. Thus, the user study setup should meet the following requirements:

1. It should guarantee that the measurement results are not biased by prior knowledge of the subjects.
2. It should require using multiple interaction devices in a consecutive and parallel order.
3. It should be applicable to all three setups.
4. It should require a dialogue between both subjects.

In an architectural design task, users had to do distance measurements in a floor plan and to integrate sketches and symbols into it (Figure 3.5, right). The users had neither prior experience in SDG, TUI, and tabletop systems, nor in architectural design. All subjects (4 female and 15 male) had a background in mechanical engineering. The tasks were as follows:

1. Measure the distance from a position in the floor plan to the nearest exit. The subjects had to mark the escape route in black and to note down its length on a piece of paper (Figure 3.5, right).
2. Measure the rooms' surface area. Following given rules, the subjects had to mark the positions of the needed fire extinguishers using red color.
3. Measure the line of sight. Within a certain range, exit signs should be visible. Using green color, the subjects should mark the positions of exit signs.

From the experiments, answers to the following questions were expected:

1. Does the new technology really allow simultaneous interaction?
2. How good do the TUIs support teamwork?

There were some hypotheses about this user study prior to performing it:

- R1: Personal tools like the pen are not shared
- R2: Efficient teamwork is characterized by sharing unique tools like the ruler and the notepad
- R3: The new system's higher responsiveness should reduce the overall completion time

In order to find answers to these questions, devices' position log was considered. The overall tabletop surface was clustered into smaller regions, for which the devices' placement frequency was analyzed. For the two pens, the results are presented in Figure 3.8.

The two graphs show the spatial probabilities of the pens. Since the task required drawing symbols, sketching lines, and entering numbers, the pens are used all over the

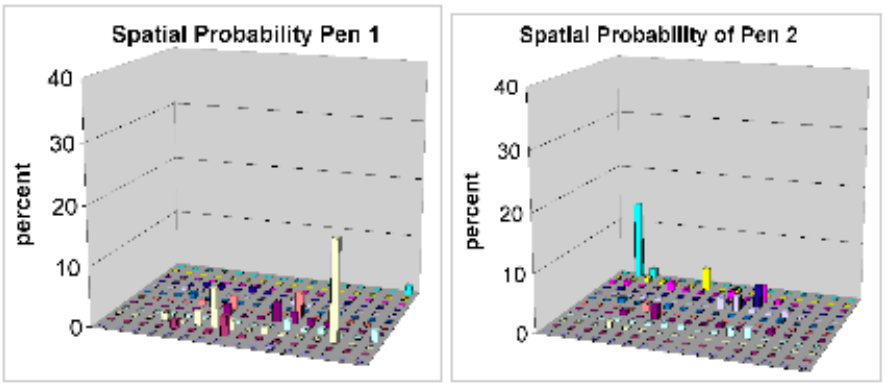


Figure 3.6: Position logs of the two pens

tabletop's surface. However, there are two significant peaks, which result from the resting position. Since all subjects were right-handed, they typically place their dominant hand together with the pen in a comfortable resting position while they are listening to the other partner. The graphs also show the face-to-face condition of the user study, in which the participants were placed at the table's long side. Measuring the ruler's positions on the table gives quite a different result (see Figure 3.7)

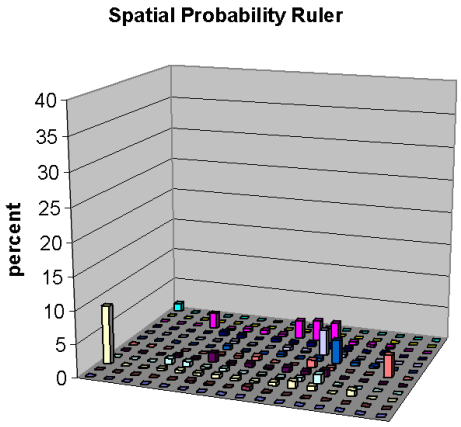
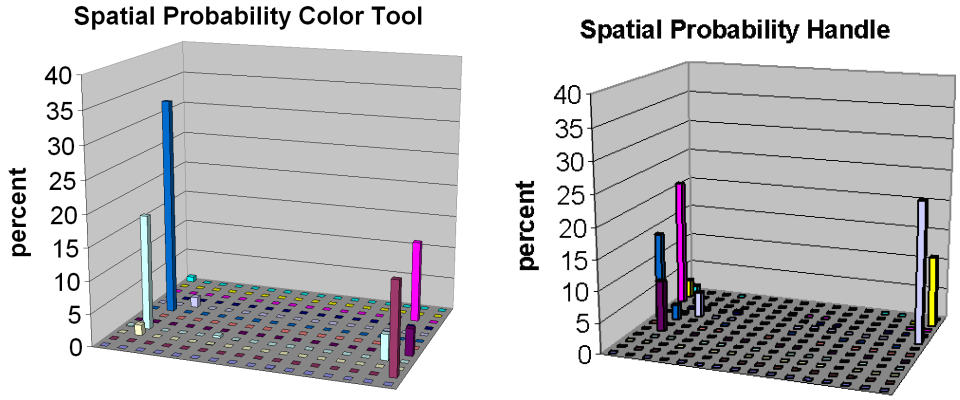


Figure 3.7: Position log of the ruler

As expected, the ruler was also used all over the tabletop's surface. However, since only one ruler was available, it was not "captured" by one of the participants, meaning that is was kept in hand and placed in a comfortable resting position. Instead, it was placed in a more neutral position (small peak on the left side of the graph) once it was not in use. Another, very different behavior resulted for the positions for the handle and the color tool (see Figure 3.8)

It is clearly visible from Figure 3.8 that these tools were not used for direct interaction



(a) (b)

Figure 3.8: Position logs of handle and color tool

with the task's floor plan, but much more rarely, e.g. for changing the pen's color or for taking notes on the notepad (handle tool). For the rest of the overall task time, these devices were placed at a position where they do not hinder the direct interaction with the floor plan. While this result seem feasible for the color tool, it is more difficult to explain for the notepad, since the task required taking notes. Thus, a more randomly distributed position log for the handle is expected . However, the users quickly recognized during the task that they also can take notes directly on the floor plan and thus did not use the handle tool anymore.

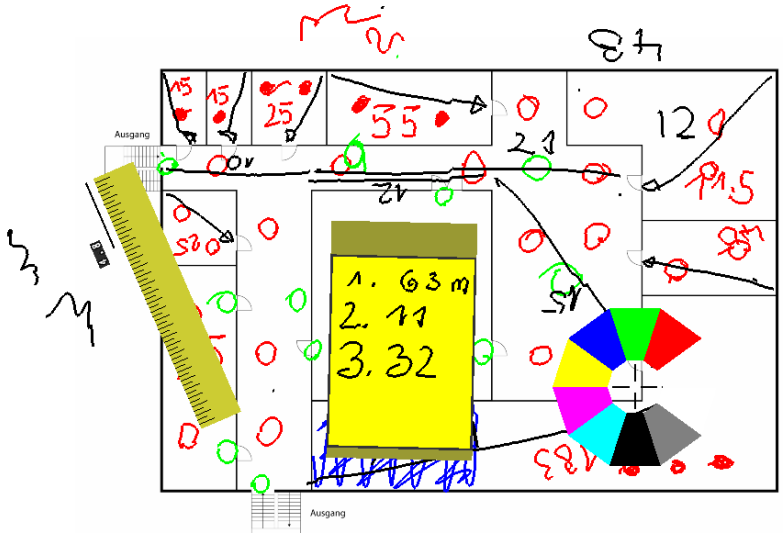


Figure 3.9: A screenshot of the completed task

A screenshot of the completed task (see Figure 3.9) also shows that users closely worked together. This can be seen by the orientation of the numbers, which are oriented to the

both long sides of the table almost to the same amount.

From the above findings, hypothesis 1 could be confirmed. Users immediately noticed (and probably also expected) that they have their personal pen for making annotations to the floor plan. It was also noticed that users never released the pen, even when they interact with other tools like the ruler.

Hypothesis 2 could only be partly confirmed. Although the ruler was frequently shared among the user, this does not hold true for the notepad, which was rarely used. This was mainly due to the fact that users quickly learned how to take "personalized" notes by writing direct on the floor plan.

Hypothesis 3 could not be confirmed, since measuring a significantly shorter completion time was not practical. This can be explained by the following reasons:

- First, the multi-state devices were not used as frequently as expected (in particular the notepad and the color tool)
- And second, there were no time-critical elements in the task that could significantly hinder the user to complete the task. However, users stated some disturbing effects when moving the TUI and then noticing a significant delay of the underlying GUI, for example when moving the transparent Plexiglas ruler and waiting for the underlying yellow GUI to also be placed again under the new position of the TUI.

3.3 Restrictions and Improvements

One of the limitations of current active TUI tabletops is that the refresh rate of tangibles is tightly coupled with the maximum number of states each TUI may have. In other words, the more states TUIs might have on the table, the lower the refresh rate of the table would be. This is due to the fact that ID, state bits, and position information are all treated as one entity, thus observed and processed with one sensor (either camera or semiconductor IR receiver), and in one process. Another problem is that the latency of detecting the existence of new object or removal of an existing object depends on the number of the TUIs.

Thus, these limitations should be improved. Mainly, the goal is to design low-latency active devices (that is, devices with more states), without reducing the high refresh rate of the system. Moreover, a cost-effective solution is preferred.

4 Dual-Mode Infrared Tracking

Dual-Mode Infrared Tracking is a new approach to overcome current restrictions of time-multiplexed IR tracking of tangibles on tabletops. This thesis suggests a distinct way to combine a low-cost camera for position detection and a low-cost IR receiver for state detection of each device, in what is called a dual mode approach.

4.1 Principle of operation

4.1.1 Basic idea

As already mentioned in Chapter 2, the bottleneck of the tracking technology is the camera. By analyzing the tasks performed by the tracking system (the camera, the central controller hardware, and the software on the PC), the system can be divided to different subsystems, each performing a set of tasks, as shown in Table 3.1

	Task	Subsystem responsible for performing the task
1	Observing the position of each device	Camera
2	Preparing the camera observations for PC	Central controller
3	Analyzing the camera observations	PC software
4	Reading the states of each device	Camera
5	Synchronizing the camera with the devices	Central controller
6	Reading the ID of the devices	Camera

Table 4.1: List of tasks performed by the tracking system's components

With a closer look at table 4.1, it can be seen that the camera performs two different tasks of observing the device's positions, and receiving their state information. With utilizing the design principle of Separations of Concern (SoC), camera tasks can be divided into the following:

1. Tracking the devices, i.e. detecting the position of the devices
2. Reading the state of each device
3. Identifying the devices, i.e. recognizing their ID

In other words, the camera is responsible of transferring all device's information: position, state, and ID.

Since transferring the position information is mainly analyzing a set of pictures, it means that a camera is needed to perform this task. But, for reading the state of the

devices, any wireless communication technology should work, since the task is simply to receive a set of bits from the devices.

Recognizing a device's ID is inherently different from the other two functionalities: ID is used not only to identify a device, but also to make the relation between its position and state. Thus, it cannot be communicated on a separate medium. Hence, both mediums which transfer state and position should also transfer the ID. This is represented in Figure 4.1[1].

ID	Position	State
1	(X_1, Y_1)	$State_1$
2	(X_2, Y_2)	$State_2$

(a) QualiTrack

ID	Position
1	(X_1, Y_1)
2	(X_2, Y_2)

ID	State
1	$State_1$
2	$State_2$

(b) Dual-Mode

Figure 4.1: A sample data model of device's information in QualiTrack versus dual-mode technology

4.2 System requirements and design parameters

In order to make reasonable decisions regarding the design of the system, the set of system requirements should be defined.

Studying the requirements for a complete interactive solution requires performing thorough user studies, defining preferred form factors, recognizing the number of (simultaneous) users, and many other application specific factors.

Since the focus of this thesis is the tracking technology of infrared tangible tabletops, only parameters that directly affect tracking of the devices need to be investigated. Thus, instead of doing new user studies to define all system parameters (e.g. physical shape of the table, type of the display, ...), it is enough to use the available information from current tabletops (in particular, QualiTrack) and user studies, like the one presented in 3.2.

Although this approach limits the number of design parameters and constraints, it also leaves some parameters undefined. Thus, reasonable assumptions should be made for defining such parameters.

Taking into account the design constraints of the system, namely the tracking frequency of position and orientation, and the number of the state information that should be delivered in a unit of time, the following decisions are made:

1. Since in a typical usage scenario, moving or rotating a tangible is used to rotate or move a visual object in the GUI (Graphical User Interface), tracking the position and orientation should be fast enough for a smooth visual representation. A camera with a frame rate of 60 frames per second delivers such a speed, since human eye can see motions smoothly at such frame rate. Moreover, since the size of the table's

[1]Interestingly, it resembles applying a second normal form (2NF) on a relational data model

	Parameter	Value	Reason
1	Maximum number of devices being used simultaneously	5	User studies on QualiTrack
2	Table size	700mmx500mm	User studies, similar products, available form factors
3	Number of states per device	2048	Desired for complex devices, like pressure sensitive pen, analog inputs, etc
4	Display frame rate	60 Hz	Available on commercial projectors

Table 4.2: Design Parameters

display is 700mmx500mm, a resolution of 640x480 pixels provides a linear resolution of less than 1.1mm, which is enough for most of the applications.

2. For transferring the state bits, any communication medium capable of a transfer rate of 20 kbps is enough. A wide variety of different wireless technologies can provide such speed. But since infrared communication is already used in the position tracking subsystem, same technology can be utilized in order to reduce complexity and cost of introducing a new communication medium and technology to the system.

4.3 Theory of operation

4.3.1 Tracking position

Much like in QualiTrack, the devices and the camera are synchronized. Moreover, each device is assigned to a specific time slot, which is equal to its hardcoded identification number (ID). In contrast to MightyTrace, all devices emit IR light on each frame, except on their assigned time slot, which is equal to their ID (see figure 4.2).

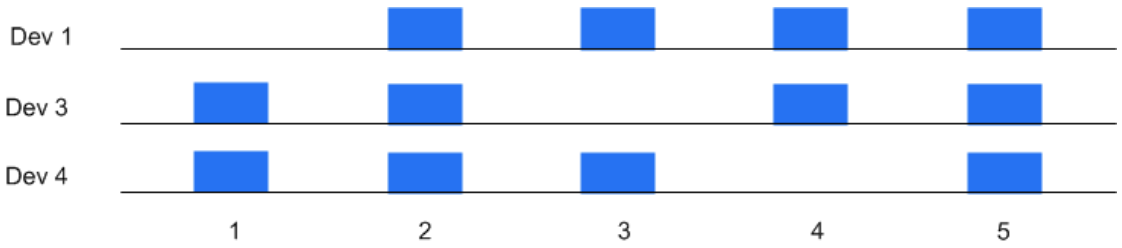


Figure 4.2: Position information transfer example with M set to 5 and with devices 1, 3, and 4 (rows) present. A cycle of 5 frames (columns) is shown. Blue cells are IR flashes sent by the devices to be detected by the camera.

Since the camera sees all the devices in every frame except one, the average update rate of the system can be calculated using formula 4.1.

$$AverageUpdateRate = f * \frac{M-1}{M} \tag{4.1}$$

where f is camera frame rate (Hz) and M is the maximum number of devices simultaneously on the table (which is preconfigured and constant in the system).

Camera frames are indexed in cycles of M frames. Since newly placed devices wait for up to one cycle to start IR transmission, the maximum setup delay equals $\frac{M}{f}$. Thus, using the dual mode method, the number of devices neither reduces the system's update rate, nor increases its lag. Only the setup delay will be negatively affected. See figure 4.3.

Thus, each device ID and position can now be identified.

4.3.2 Identifying states

Each device transmits its state information using its IR LED between two synchronization signals, i.e. the speed of transmitting the state information is significantly higher than the speed of the camera (see figure 4.4).

This is feasible since the state information is read by a simple IR receiver and not by the camera. The interval between two consecutive camera frames is further divided into M sub-frames. Within each sub-frame, only the corresponding device sends its state information. The bit rate, R, of the employed sensor can be up to 22 kbps. Hence, with the camera exposure time, e, and f and M as defined above, the maximum number of state bits per device equals R ((1/f - e)/M). For example, with M set to 5, f at 60 Hz, and e at 10 ms, each device can transmit up to 29 bits of state information, allowing more than half a billion states per device.

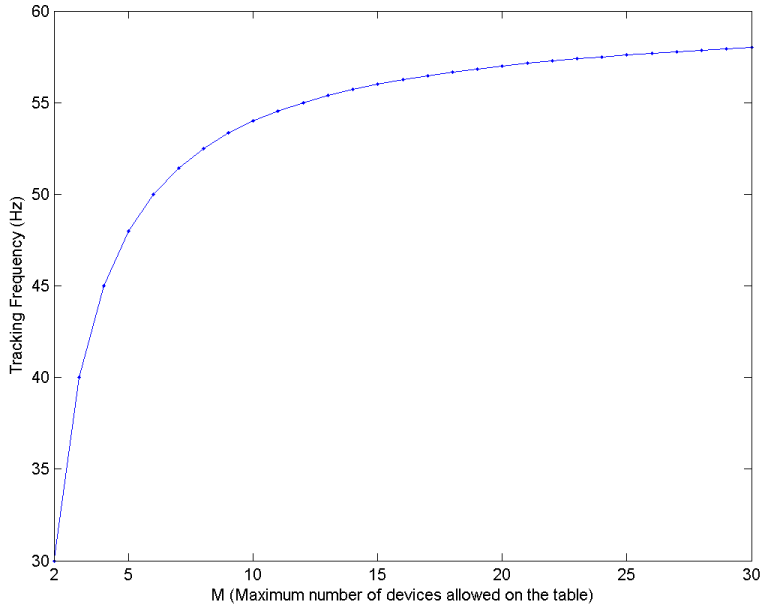
4.4 Hardware design

Based on the design choices, the system is divided into the following subsystems:

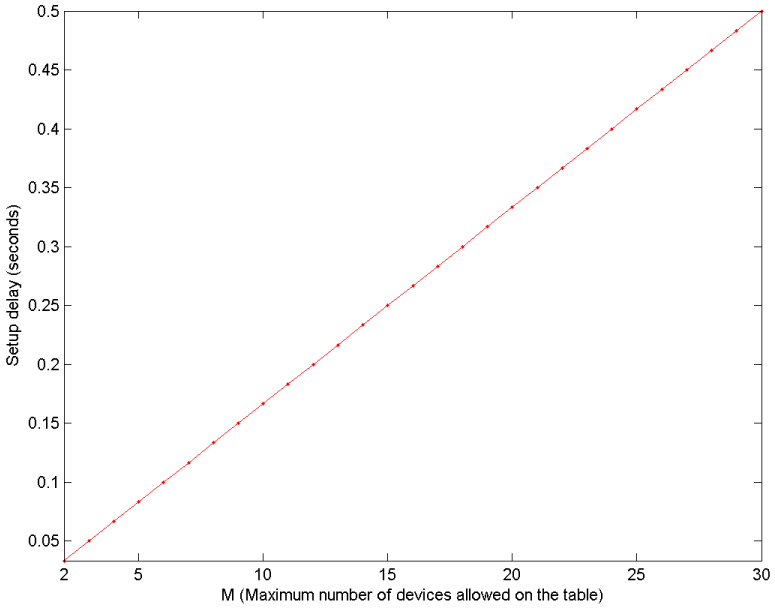
1. Camera
2. central controller
3. IR Flash, used for synchronization
4. IR receiver module

4.4.1 Camera

In order to lower the total cost of the system, an affordable camera, which not only satisfies the requirements mentioned in Section 4.2, but also provides standard hardware interfaces and protocols, and well supported software libraries and API (Application programming interface) should be selected.



(a)



(b)

Figure 4.3: Effect of different M configurations on a table with a 60 fps camera on (a) update rate and (b) setup delay

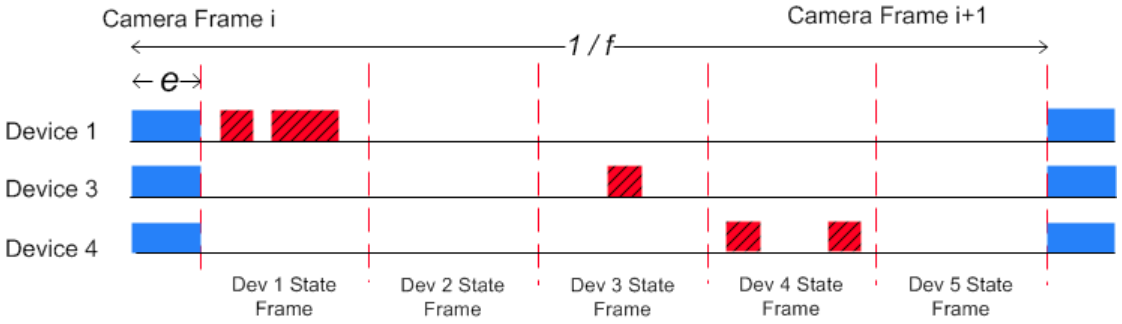


Figure 4.4: State information transfer example with M set to 5 and with devices 1, 3, and 4 (rows) present. One interval between camera frames i and i+1 is shown. Blue cells are IR output of TUIs that are detected by the camera, used for tracking the positions; red cells are IR outputs of the TUIs that are seen by the IR receiver, used for reading the states.

Specifically, it is important that the camera supports external hardware triggering, since synchronizing the camera shots with the IR receiver module is crucial for the system to function correctly. Moreover, its image sensor should be capable of detecting infrared light.

These limitations and desired attributes of the camera lead to selecting a **PointGrey FireFly FMVU-13S2C** camera. The camera is equipped with USB 2.0 connection for sending image data, and a GPIO (General Purpose Input Output) connector for sending and receiving commands to and from the camera.

The camera's image sensor is *a Sony IMX035* complementary metal–oxide–semiconductor (CMOS). This sensor has a **Quantum Efficiency (QE)**[2] of 15% for 840nm wavelength, which is low, compared to other available camera sensors, as shown in figure 4.5. But it comes with a 12-bit Analog to Digital Converter (ADC), which is 4 times more sensitive than most of the presented sensors. Moreover, the infrared LEDs used on the devices are on their maximum intensity. In total, the sensor delivers enough sensitivity level for tacking the TUIs.

It should be noted that, as many other color cameras, FireFly FMVU-13S2C comes with an IR cut-off filter installed, with a transmittance graph of figure 4.6. Thus, the filter should be remove before and be replaced by an IR pass filter. This results in an inexpensive infrared camera, which satisfies the system's requirements.

4.4.1.1 Trigger and strobe signals

The camera's GPIO pins can be configured to perform different functionalities. Among these, trigger input and strobe output can be used to synchronize the camera with another device: receiving a signal on the trigger input initiates image acquisition, and at the beginning of image integration a strobe pulse is generated.

[2]The percentage of photons hitting the imaging surface at various wavelengths that can produce a charge. It is a measurement of the imager's electrical sensitivity to light.

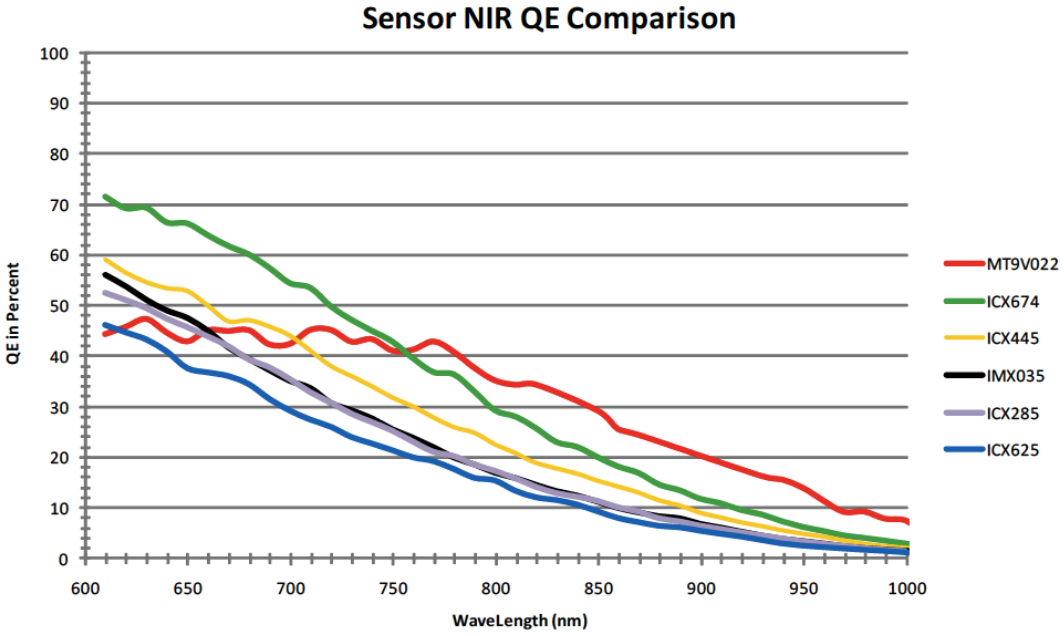


Figure 4.5: Near InfraRed QE comparison of different image sensors

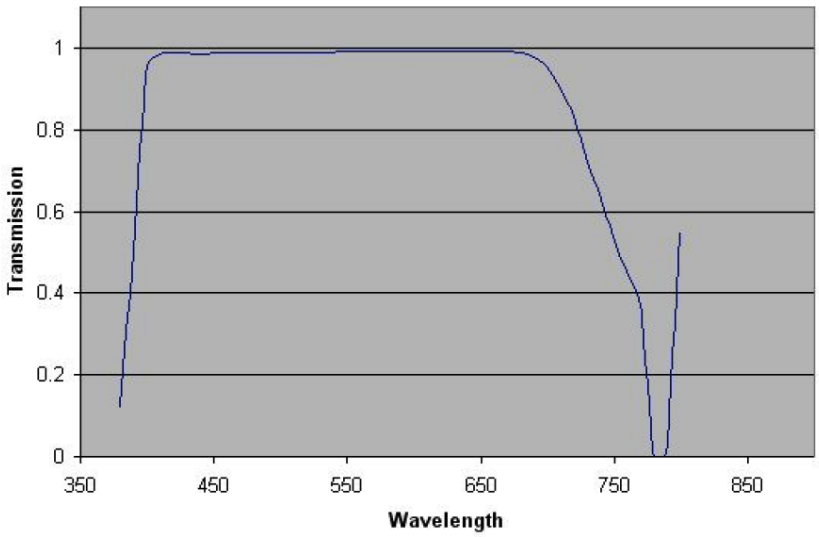


Figure 4.6: Infrared cut-off filter installed on PointGrey FireFly FMVU-13S2C camera

Because of limitations of the camera sensor[3], using an external trigger halves the maximum frame rate of the camera. In other words, the camera's frame rate would decrease to 30 frames per second. On the other hand, using the camera in free run mode allows using it in its highest frame rate. Thus, instead of using the trigger input, strobe output can be used to synchronize the camera with all the other components, without reducing its frame rate. In a way, the camera acts as the main system clock, generating a pulse whenever an image integration phase is initiated.

4.4.2 Central controller

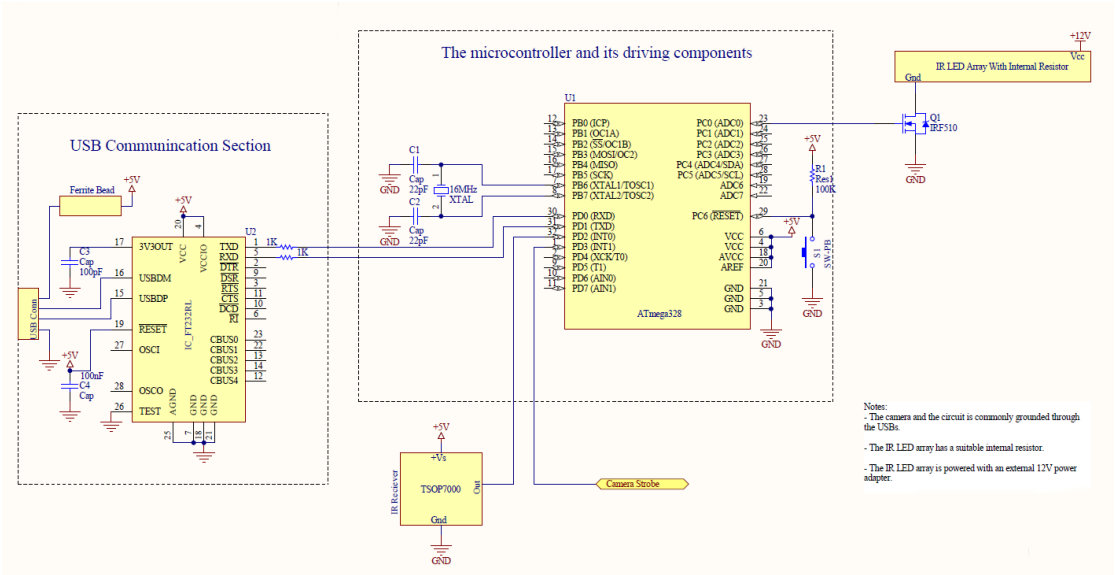


Figure 4.7: Electronic circuit for the central controller

The central controller is the heart of dual-mode IR tracking system. It performs three main tasks:

1. Synchronizing all subsystems: central controller is responsible of triggering the flash and synchronizing it with the camera
2. Receiving the state bits
3. Sending tracking, ID, and state information to the PC

This task list leads to selecting an embedded computer system. Since the camera controller's operating frequency is low, the selection of a processor architecture is not based

[3]The camera has a rolling shutter sensors. These sensors work differently in external trigger mode and free running mode: in free running mode, the read out and image integration happens synchronously (with a small time shift between the two operations), while in external trigger mode, the image data is read out only after the whole image is integrated, while . Thus, an additional frame is needed for read-out, decreasing the total frame rate to half.

on the performance. Parameters like availability of the device and tool chains, range of supported protocols and peripheral devices, are the deciding factors. Taking these into account, the design is decided to be based on a simple 8- bit AVR microcontroller, namely an ATMega328.

The central controller is divided into a number of subsystems, each facilitating one of the central controller's tasks, as depicted in figure 4.8.

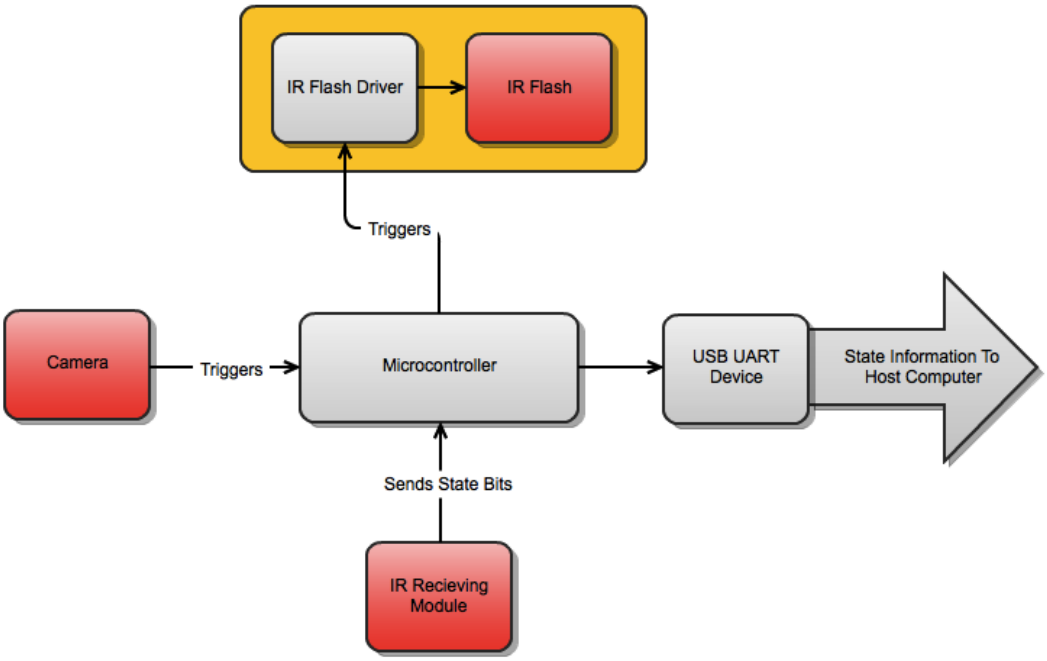


Figure 4.8: central controller

4.4.2.1 IR Flash Driver

As mentioned in 4.4.3, four 3.2 Watts infrared bars are used as flash, which means the power consumption of the flash is 4x3.2 = 12.8 Watts. Moreover, since the driving voltage of the bars are 12V, the current that passes though the flash is 1.07A. For switching such a current, a driving circuit is required, since the highest current that the microcontroller can directly switch is 40 mA.

Among many components that may serve this purpose, the most available one, an IRF510, is chosen. This N-channel Power MOSFET is capable of switching up to 5.6 A. Moreover, since it is a Field Effect Transistor, it can be directly driven by the microcontroller (It has a Gate-to-Source current of 100 nA, and a threshold voltage of 4V).

4.4.2.2 IR Receiver Module

The IR Receiver Module is responsible of reading the state information from the devices. The module is a TSOP7000, and infrared receiver, integrating a photo detector and a preamplifier. It works at 455 KHz, and uses pulse-code modulation (PCM), delivering a noimal data rate of 20 kbps[4].

4.4.3 IR Flash

The flash serves as the synch signal for all TUI devices: it synchronizes the devices with central controller. This flash will be placed below the table surface, in a way that it doesn't block the projector's light. Since TUI devices may be placed in different position on the table surface, it should be certain that the flash pulses cover the whole surface, so that all devices can receive the flash pulses.

A widely used component for such application is an infrared LED bar, which is an array of infrared LEDs, and their driving circuits, geometrically arranged in a row. To be sure about the coverage of the flash light, one bar on each side of the table should be installed, hence a total of 4 infrared bars.

Moreover, a 850nm infrared light bar is selected, which delivers 593 milliwatts of brightness, with a power consumption of 3.2 Watts. It requires a 12 Volts DC power supply, which is provided by an external power source.

4.5 Hardware Implementation and Measurements

In a first prototypical setup, the system's performance was measured (see 4.9). It shows the ATmega328 8-bit AVR microcontroller and the TSOP7000 IR receiver connected to it. Further, the IR emitter is shown. The analog read-out value of a potentiometer is converted to an 8 bit digital signal using the microcontrollers' internal analog to digital converter, which is then sent to the IR emitter.

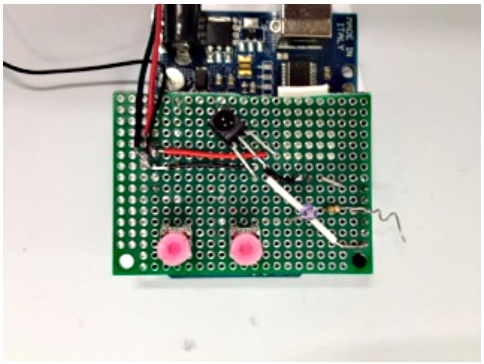
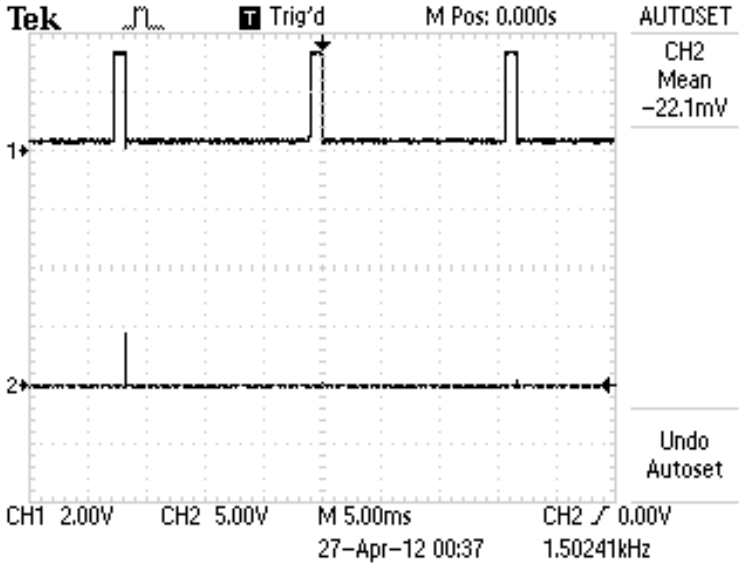


Figure 4.9: Prototypical setup for measuring the system's performance

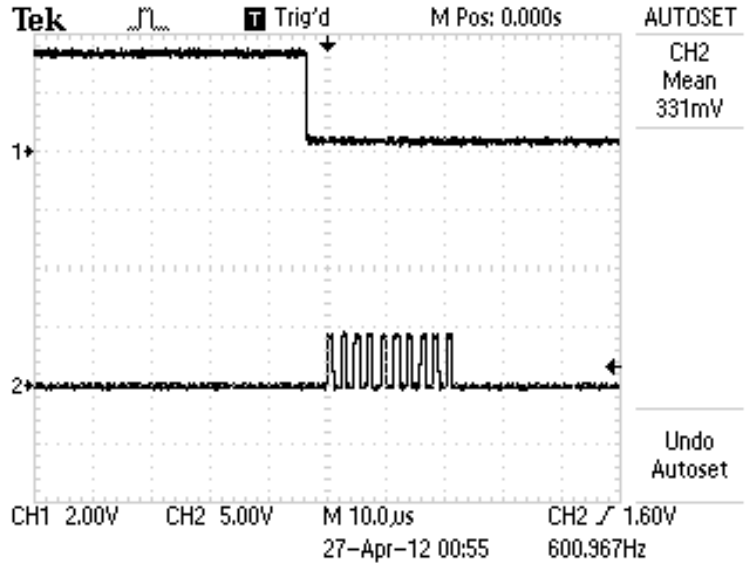
[4]The real data rate acheived in the system was up to 22 kbps.

Using an oscilloscope, the signal at the receiver was measured together with the camera's triggering signal (see 4.10). In Figure 4.10a, the upper signal shows the triggering signal for the camera, running at 60 Hz. The lower signal is the burst (trigger), sent via the IR flash array to the devices. However, this burst for the devices cannot be seen here since it is much shorter (i.e. only a spike can be seen in channel 2). Thus, a higher temporal resolution is chosen (t = 10 µs/unit) in order to resolve the burst (see Figure 4.10b). Now, it can be seen that a burst consists of 10 pulses (1 pulse corresponds to the required carrier frequency of 455 kHz), lasting 22 microseconds. After such a burst, the system waits for another 28 microseconds before sending the next burst (in case of a device's response). Figure 4.10c shows such a device response, consisting of 7 bits.

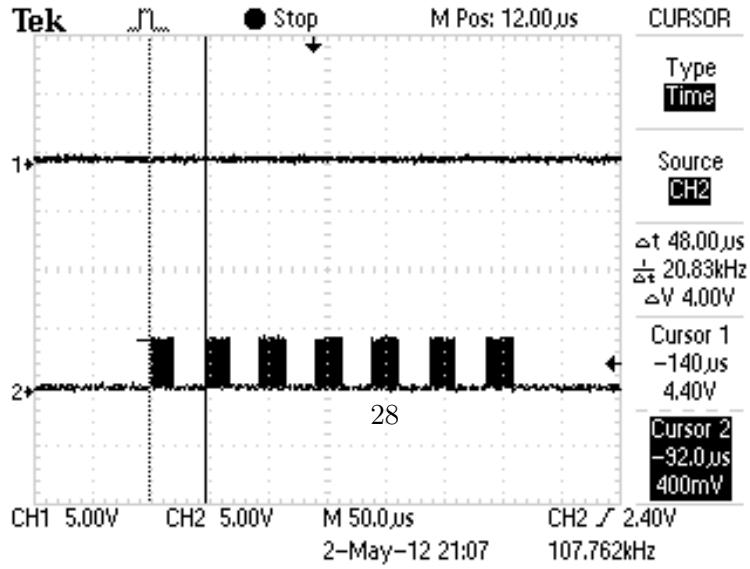
In conclusion, the technical system is capable of transferring a sufficiently large amount of states without reducing the overall transfer speed of the system. Thus, it is possible to realize devices for the more complex applications.



(a)



(b)



(c)

Figure 4.10: Measurement results for the prototypical setup

4.6 Software design

4.6.1 Main Controller's firmware

The central functionality of the main controller is performed after a strobe from the camera is received on the microcontroller interrupt pin.

Algorithm

1. Set FrameNumber to zero.
2. On receiving a strobe signal from the camera,
   a) Send the data array to the USB output.
   b) Wait for 2 milliseconds, and send a sync signal[5] by switching the IR LED on.[6]
   c) Set fastFrameCounter to zero.
   d) Set FrameNumber = (FrameNumber + 1) MOD FrameCycle.
3. Read the output of the IR receiver (on Port D, pin #1) every 50 microseconds. Assign it to data[fastFrameCounter] in a MSB manner.
4. Increase fastFrameCounter every 2 milliseconds

4.6.2 Device's firmware

Algorithm

1. On receiving two LOW bits on IR receiver output (PORT D, pin #0), set FrameCounter to 0 (Syncing first frame cycle).
2. On receiving one LOW bit on IR receiver output (syncing all other frames),
   a) If FrameCounter == -1, do nothing
   b) Else, if FrameCounter == ID, turn off the LED, else, turn it on.
3. If fastFrameCounter is equal to ID, then send the user input using IR LED (a total of 25 bits).
   a) Send a bit every 50 microseconds, using the mentioned burst signals. Send ***data[fastFrameCounter]*** in a MSB manner.
4. Increase ***fastFrameCounter*** every 2 milliseconds

[5] If FrameNumber is zero, sync signal should be two bits, otherwise it is one bit.

[6] Switching on here means sending a burst of 10 pulses, each with a period of 2.5 microseconds, resulting in a total burst of 25 microseconds, and waiting for 25 more microseconds: This sends a bit. Thus, each bit takes 50 microseconds to send, or a 20 kbps communication

4.6.3 Camera Tracker

The camera tracker is the software running on the host computer. It collects the information from the camera, as well as the central controller, combines them together, extract ID, position and state information from the raw data, and encapsulates them into a single object.

4.6.3.1 Algorithm

1. Read camera frame.
2. Detect bright points in the frame. For each point, assign it to the object which is a candidate for being the owner of the point (Simply, if the detected point is close enough to one of the devices that were previously recognized, that device is the owner of the point). If such device doesn't exist, it means a new device is added.
3. When FrameCount number of frames are detected for a device, check if it is valid using the following approach:
   a) If there is only one blank frame in the device's frames, set device identification number to that frame number. Increase the success counter for statistical purposes.
   b) Else, there is an error in the system. Simple, increase error counter. (Further error correction algorithm might be necessary when increasing the number of devices, or decreasing the power consumption of the devices)
4. Always check if there are duplicate devices on the table (devices with the same ID). In that case, an error should be logged.

A flow chart of the algorithm is represented in figure4.11.

Pseudocode

```
frameNumber = 0;
DetectedDevices = new List<Device>();
int DuplicateDeviceError = 0;
while(true)
{
        newFrame = ReadCameraFrame();
        ListOfPoints = FindBrightPoints(newFrame);
        for each Point P in ListOfPoints
        {
                bool deviceFound = false;
                for each Device D in DetectedDevices
                        if (BelongsToDevice(P,D))
                        {
```

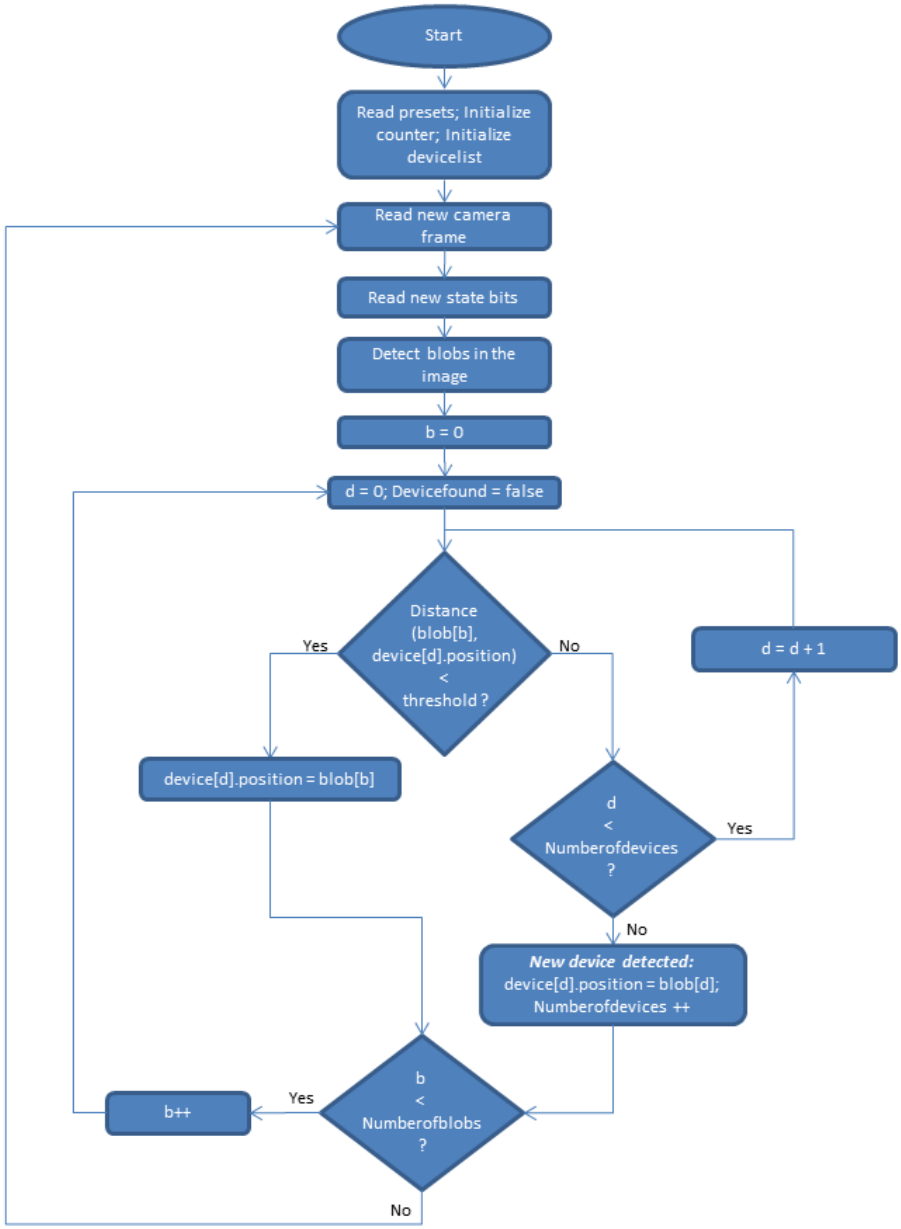


Figure 4.11: Camera tracker flowchart

```
                                        D.AddNewPoint(P,frameNumber);
                                        deviceFound = true;
                                        break;
                                }
                        if (!deviceFound)
                        {
                                D = NewDevice();
                                DetectedDevices.Add(D);
                                D.AddNewPoint(P,frameNumber);
                        }
                }

                frameNumber = (frameNumber + 1) % FrameCycle;
                CheckForDuplicateDevices();
}

void OnUSBDataReady()
{
        int* data = USB.ReadData();
        foreach (Device D in DetectedDevices)
                D.Data = data[D.ID];
}

void CheckForDuplicateDevices()
{
        for (int i=0; i<DetectedDevices.Length; i++)
        {
                for (int j=0; j<DetectedDevices.Length; j++)
                        if (i!=j)
                                if (DetectedDevices[i].ID ==
                                    DetectedDevices[j].ID)
                                        DuplicatedDeviceFound();
        }
}

void DuplicatedDeviceFound()
{
        DuplicateDeviceError ++;
}

Device::AddNewPoint(Point P, int frameNumber)
{
        this.PointList[frameNumber] = P;
        this.FrameList.Add(frameNumber);
```

```
    // if the final frame is reached    , check for Device
       Identification frame
        if (this.FrameList.Length == FrameCycle)
        {
                int numberOfBlankFrames = 0, int ID = -1;
                for (int i=0;i<FrameCycle;i++)
                        if (this.PointList[i] == null)
                        {
                                numberOfBlankFrames++;
                                ID = i;
                        }

                if (numberOfBlankFrames == 1)
                {
                        this.ID = ID; this.Found();
                }
                else
                {
                        this.Error();
                }
        }
}

bool BelongsToDevice(Point P, Device D)
{
        // This is the simplest approach. It should be improved,
            with velocity and acceleration comparisons.
        return (Distance(this.PointList.Last, P)<
           DistanceThreshold);
}

Device::Found()
{
        // For statistical purposes this.
        Success++;
}

Device::Error()
{
        // For statistical purposes this.
        Error++;
}
```

5 Conclusion and future work

In this master thesis, a new approach for tracking active tangible user interfaces on tabletop is presented. It was shown that, how using two different receivers, the tracking speed of the system, as well as the number f states for each TUI, can be improved.

The main advantage of such a system is that the update rate of the system is only limited bye the camera's frame rate. In other words, increasing the number of devices or the number of states doesn't deteriorate it. Another advantage of the dual IR approach is that the number of states per device could be very high: in the excess of millions of states. Moreover, the system is implemented using off the shelf components, thus the system can be realized with low-cost components.

There are still some possible improvements of the system. Specifically, the algorithm used for tracking the device positions (presented in 4.6.3) only uses the positions of the devices in order to track them. Though this approach is good enough, but including other factors such as velocity, acceleration, etc make the tracking much more solid.

Finally, since dual-mode infrared tracking improves the overall performance of a tabletop, creating more intelligent devices for such a system is easily realizable. As a result, these devices can be used to deliver a higher level of usability. Designing such devices, as well as applications for using those devices, is an important next step.

Bibliography

[1] Belcher D. Arnab G. Billinghurst, M. Communications behaviours in collocated collaborative ar interfaces. In *International. Journal of Human-Computer Interaction*, 2003.

[2] Buxton W. Fitzmaurice G., Ishii H. Bricks: Laying the foundation for graspable user interfaces. In *Proceedings of CHI 1995*, 1995.

[3] Steinemann A. Kunz A. Ganser, C. Infractables: Multi-user tracking system for interactive surfaces. In *Proceedings of IEEE Virtual Reality Conference*, 2006.

[4] Ullmer B. Ishii H. Tangible bits: Towards seamless interfaces between people, bits and atoms. *Proceedings of CHI Õ97*, 1997.

[5] Ajay Kapur Jordan Hochenbaum. Adding z-depth and pressure expressivity to tangible tabletop surfaces. 2011.

[6] Alonso M. Kaltenbrunner M. Jourda S., Geiger G. The reactable: Exploring the synergy between live music performance and tabletop tangible interfaces. In *Proceedings of the 1st International Conference on Tangible and Embedded Interaction*, 2007.

[7] Fjeld M. Kunz, A. *From Table-System to Tabletop: Integrating Technology into Interactive Surfaces*. 2010.

[8] Andreas Kunz Ramon Hofer, Thomas Nescher. Qualitrack: Highspeed tui tracking for tabletop applications. 2009.

[9] Patrick Kaplan Ramon Hofer, Andreas Kunz. Mightytrace: Multiuser tracking technology on lc-displays. 2008.

[10] Ishii H. Ullmer B. The metadesk: Models and prototypes for tangible user interfaces. In *Proceedings of UIST Õ97*, 1997.

[11] A. Wilson. Playanywhere: A compact interactive tabletop projection-vision system. In *Proceedings of the 18th Annual ACM Synposium on User Interface Software and Technology UIST 2005*, 2005.

6 Appendix: Publications

6.1 Dual Mode IR Position and State Transfer for Tangible Tabletops: As appeared in ITS 2011, Kobe, Japan

Dual Mode IR Position and State Transfer for Tangible Tabletops

Ali Alavi
t2i Lab, Chalmers TH
SE-41296 Gothenburg, Sweden

Masanori Sugimoto
Interaction Technology Lab, University of Tokyo
113-8656, Tokyo, Japan

Andreas Kunz
ICVR, ETH Zurich
CH-8092 Zurich, Switzerland

Morten Fjeld
t2i Lab, Chalmers TH
SE-41296 Gothenburg, Sweden

ABSTRACT
This paper presents a method for tracking multiple active tangible devices on tabletops. Most tangible devices for tabletops use infrared to send information about their position, orientation, and state. The method we propose can be realized as a tabletop system using a low-cost camera to detect position and a low-cost infrared (IR) receiver to detect the state of each device. Since two different receivers (camera and IR-receiver) are used simultaneously we call the method *dual mode*. Using this method, it is possible to use devices with a large variation of states simultaneously on a tabletop, thus having more interactive devices on the surface.

General Terms: Algorithms, Performance

Keywords: Active tangible devices, tabletop, dual mode, IR tracking, multi TUI

INTRODUCTION
The advent of widely available interactive tabletops, such as the MS Surface, has created high expectations among users for such systems. Today users expect a highly interactive experience when interacting at tabletops. Tangible objects emitting active infrared (IR) could be one possible way to provide a more interactive user experience with tabletops.

QualiTrack [2] delivers such an experience by providing users with active TUIs. In this system, TUIs have intuitive shapes such as brick, color-pallet, and pen, which may enable a more natural interaction style. Each of these TUIs has a set of state-triggering widgets, such as buttons on top of the bricks or a micro switch under the pen's tip. Thereby, users can interact with an object while positioning it on the surface. However, based on user feedback from studies on QualiTrack, a majority of users were not satisfied with the level of interactivity they experienced with the TUIs provided. In other words, users expect tangible objects to be much more than simple point and click devices. Thus, we decided to improve the TUIs to achieve higher user satisfaction. We observed that a significant improvement in perceived interactivity in tangible tabletops could be achieved by increasing the number of states a TUI can deliver. This enables system designers to offer more complex forms of interaction. For example, a high-end pressure-sensitive stylus may have up to 2048 pressure levels. To implement such a pressure sensitive stylus for systems like QualiTrack, the TUI needs to send 11 state bits. We could not reach this number of states on the initial system since the update rate would then drop dramatically. In this paper, we address this problem by introducing a new tracking method for use in active tangible tabletops. This method allows us to build tangible tabletops with a high number of states using low-cost components.

RELATED WORKS
Our work has been inspired by two contributions in the area of tangible tabletops using active TUIs: SmartFiducial [1] and MightyTrace [3]. In SmartFiducial, active tangibles communicate with a host computer using wireless radio frequency (RF) transmission. Object positions are detected using a visual tracking system, thus adding a further level of complexity to the TUI design, as well as the host computer. Moreover, there is always a risk of unintended interference with RF communication. Hence, we deemed this approach to be unsuitable for our project. In MightyTrace, a matrix of IR sensors is used to detect the position and state of TUIs. Each device is assigned a specific time frame during which its LED is turned on. Thus, each device can be detected unambiguously. For sending states, there are even more time slots for each device. For example, if a device has eight different states it then needs four time slots, one for detecting its position and three for sending its state bits (thus eight different states). However, the main restriction with such a time-multiplexed approach is that the system update rate is limited not only by the number of devices on the surface, but also by the number of states each device may have.

ITS 2011, November 13-16, Kobe, Japan.

DUAL MODE TRACKING

Here, we introduce a new approach to overcome current restrictions of time-multiplexed IR tracking of tangibles on tabletops. Mainly, we aim to design low-latency active devices, that is, devices with more states without reducing the high refresh rate of the system. Moreover, we are interested in a cost-effective solution. We suggest a distinct way to combine a low-cost camera for position detection and a low-cost IR receiver for state detection of each device, in what we call a dual mode approach.

Design considerations

We are interested in having a relatively small number of tangible devices on the tabletop, each having a relatively high number of states. For example, considering the physical size of a table and the states required for a high-end pressure-sensitive stylus, we may simultaneously have five styli on the surface, each with 2048 possible states or pressure levels. To meet this requirement, we use two different receivers: a camera capable of detecting positions and an IR receiver capable of detecting the states of the devices. Concerning latency requirements, two terms are frequently employed: "update rate," which is the number of positions and states being updated per second, and "lag," which is the response time of the system to user input. We introduce a third term relevant to the method presented here: setup delay. This is the time from when a device becomes present in the tracked area until it is recognized.

Tracking position and orientation

Much like in QualiTrack, the devices and the camera are synchronized. Moreover, each device is assigned to a specific time slot. All devices emit IR light on each frame, except on their assigned time slot (Fig. 1). Since the camera sees all the devices in every frame except one, the average update rate of the system equals $f(M\text{-}1)/M$, where f is camera frame rate (Hz) and M is the maximum number of devices we want to have on the table (which is preconfigured and constant in the system). Camera frames are indexed in cycles of M frames. Since newly placed devices wait for up to one cycle to start IR transmission, the maximum setup delay equals M/f. Thus, using the dual mode method, the number of devices does not reduce the system's update rate, nor does it increase its lag. Only the setup delay will be negatively affected. We can now unambiguously identify each device and its position. Device tracking uses a blob-tracking algorithm [4]. While we assume that a device has one LED source only, instrumenting a device with two or more LED sources and combining their positions can give device orientation [3].

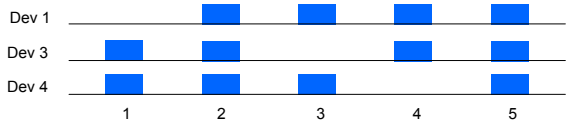


Figure 1. Position information transfer example with M set to 5 and with devices 1, 3, and 4 (rows) present. A cycle of 5 frames (columns) is shown. Blue cells are IR flashes sent by the devices to be detected by the camera.

Identifying states

Each device transmits its state information using its IR LED between two synchronization signals, i.e. the speed of transmitting the state information is significantly higher than the speed of the camera (Fig. 2). This is feasible since the state information is read by a simple IR receiver and not by the camera. The interval between two consecutive camera frames is further divided into M sub-frames. Within each sub-frame, only the corresponding device sends its state information. The bit rate, R, of the sensor we employ can be up to 22 kbps. Hence, with the camera exposure time, e, and f and M as defined above, the maximum number of state bits per device equals $R((1/f - e)/M)$. For example, with M set to 5, f at 60 Hz, and e at 10 ms, each device can transmit up to 29 bits of state information, allowing more than half a billion states per device.

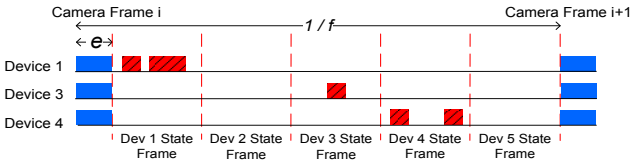


Figure 2. State information transfer example with M set to 5 and with devices 1, 3, and 4 (rows) present. One interval between camera frames i and i+1 is shown. Blue cells are IR flashes to be detected by the camera; red cells are IR flashes to be detected by the IR receiver.

CURRENT STATUS AND FUTURE WORKS

We evaluated the feasibility of the proposed method by implementing its essential subsystems. Particularly, we implemented the IR receiver and changed the QualiTrack TUIs to send state information using our dual mode method. We also investigated whether the battery operated TUIs allow us to send signals powerful enough to be detected by the IR receiver, considering the distance between the TUIs and the sensor. Our findings show that it is feasible to implement a tabletop using our new method. A next step in this project will be to implement a complete tabletop using the method presented here.

REFERENCES

1. Hochenbaum, J., Kapur, A. (2011): Adding Z-Depth and Pressure Expressivity to Tangible Tabletop Surfaces, *Proc. ACM NIME 2011*, 3 pages.
2. Hofer, R., Nescher, T., Kunz, A. (2009): QualiTrack: Highspeed TUI Tracking for Tabletop Applications, *Proc. INTERACT 2009*, Springer.
3. Hofer, R. (2011): Tracking technologies for interactive tabletop surfaces, PhD Thesis, *ETH Zurich*.
4. Schöning, J., Hook, J., Bartindale, T., Schmidt, D., Oliver, P., Echtler, F., Motamedi, N., Brandl, P., Zadow, U. (2010): *Building Interactive Multi-touch Surfaces, Tabletops - Horizontal Interactive Displays*, Springer.

6.2 Multi-State Device Tracking for Tangible Tabletops: As appeared in SIGRAD 2011, Stockholm, Sweden

Multi-State Device Tracking for Tangible Tabletops

Ali Alavi[1], Brice Clocher[1], Allen Smith[1], Andreas Kunz[2], and Morten Fjeld[1]

[1]t2i Lab, Chalmers TH, SE-41296 Gothenburg, Sweden
[2]ICVR, ETH Zurich, CH-8092 Zurich, Switzerland

Abstract

On tangible tabletops, Tangible User Interfaces (TUIs) can signalize their identity, position, orientation, and state by active infrared light. This provides rich interaction capabilities in complex, dynamic scenarios. If TUIs have to transfer additional high-resolution information, many bits are required for each update. This has a negative impact on the overall update rate of the system. In the first part of this paper, we present an in-house map application where interaction with time-dependent contour lines may benefit from high-resolution TUI states. Prototypical TUI concepts such as slider, ruler, and dials further motivate the benefit of high-resolution tracking. In the second part of the paper, we depart from a device tracking overview and then show how tangible devices for tabletops typically use infrared (IR) emitters and a camera to send information about their position, orientation, and state. Since transferring many additional information bits via a normal camera-based tabletop system is not feasible anymore, we introduce next a new system setup that still offers a sufficiently high update rate for a smooth interaction. The new method can be realized as a tabletop system using a low-cost camera detecting position, combined with a low-cost infrared receiver detecting the state of each device. Since both kinds of sensors are used simultaneously we call the method "dual mode." This method combines a camera-based tracking with the possibility to transfer an almost unlimited amount of states for each device.

Categories and Subject Descriptors (according to ACM CCS):
General terms: Algorithms, Performance
Keywords: Active tangible devices, tabletop, dual mode, IR tracking, multiple TUI

1. Introduction

The advent of widely available interactive tabletops has created high expectations among users for such systems. Tangible tabletops where active devices (TUIs) are tracked to inform about their identity, position, orientation, and state can provide rich interaction within complex, dynamic scenarios. InfrActables [GSK06] delivers such an experience by providing users with active TUIs. In that system, TUIs have form factors such as pen, handle, ruler, and color tool, which may enable a more natural interaction style. Each of these TUIs has a few state-triggering widgets, such as buttons on top of the bricks or a micro switch under the pen's tip. Thereby, users can control states while positioning the TUI on the surface. Informing about TUI states over a large range at a high-resolution requires sending many bits for each update and comes at the cost of system update rate. However, tracking of TUIs has so far been limited to the spatial and temporal resolution of the camera CCD chip. Future scenarios where TUIs control dynamic high-resolution parameters of dynamic map-based scenarios, economic simulations, or science education are promising [KF10]. In such application, TUIs with dynamic high-resolution input streams will enable system designers to use richer forms of interaction. For instance, a slider, ruler, or dial equipped with a potentiometer can send its input value (sampled though an A/D converter) offering up to 2048 adjustment levels. To implement such functions for systems like InfrActables, a TUI needs to employ a much higher number of state bits than today. We could not utilize this number of states on the initial system because the update rate would then drop dramatically. In this paper, we address this problem by introducing a new additional state detection method for use in active tangible tabletops. This method allows us to build tangible tabletops with a high number of states using low-cost components.

The first part of this paper presents an existing map-based application where interaction with time-dependent contour

Figure 1: *A tangible tabletop map-based application for TUI and pen-based interaction with Google Maps.*

lines may benefit from high-resolution TUI states. Previously researched TUIs such as slider, ruler, and dials can be expected to work with our system. Besides being a research contribution in and of itself, our application serves also as motivation to research improved multi-state device tracking. In the second part of this paper, we first give an overview of alternative forms of tabletop tracking before we focus on the tracking of actively emitting devices. The second contribution of this paper, an improved multi-state device tracking method, called "dual mode", is then presented. The paper summarizes the current status and indicates potential future work.

2. Tangible Tabletop Map-based Application

Users involved in time-critical planning with interactive maps sometimes use physical tools like ruler, dials, and pens in order to share knowledge with each other and collaborate in creating a common operational picture. Previous research has demonstrated that tangible tabletops can help in these tasks [PIA09]. For the purpose of crisis resource management, we have built an interactive table with tangible devices and an information visualization framework prototype. The framework allows creating crisis management scenarios using Google Maps-based Flash applications. To interact with this application, we propose actively emitting TUIs such as slider, ruler, and dials. While the design and use of these devices has been proposed in related projects [GSK06] [SJG*06] [WWJ*06], we consider the tracking method presented next as critical in making the use of such devices successful.

One of the most important features of a tangible crisis management application is believed to be time-dependent shape, or contour line, editing. Indeed, crisis management experts often draw shapes on paper maps, for instance, in order to represent the spreading of a fire. Those shapes are

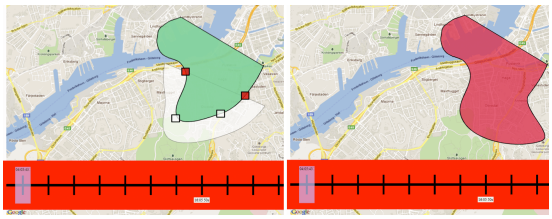


Figure 2: *Map interaction screenshot: Shape control is done by moving the nodes of a parametric curve; time control employs time-line at the bottom.*

associated with thematic, spatial, and temporal content. In the case of a fire, a shape may represent an area that is burning at a specific time. In one related example, Igarashi et al. [IMH05] presented algorithms and applications where users can move and deform a two-dimensional shape without manually establishing freeform deformation (FFD) domain beforehand. Inspired by this work, one of our current implementations employs a so-called parametric curve description. Using Flash parametric curve libraries, we have been able to implement from a rather high level of abstraction (Figure 2). When expert users receive data about the development of a phenomenon, they often model this development using time-dependent shapes. A first problem encountered when drawing shapes on a map is how to associate temporal and thematic content with a shape. In our system this raises two issues: firstly, how tangible interaction can make shape and time control precise but still easy and intuitive, and, secondly, how a user can modify shapes while still being able to keep track of their associated temporal, spatial, and thematic contents. As for the first issue, we conjecture that TUIs such as slider, ruler, and dials may benefit shape and time control. Figure 3 shows some prototypical uses of multi-state devices such as dials and a frame. Tracking TUI states with values assuming values over a large range at a high-resolution requires sending many bits for each update without compromising the system update rate. Achieving this without making compromising on system update rate is the focus for the remainder of this paper.

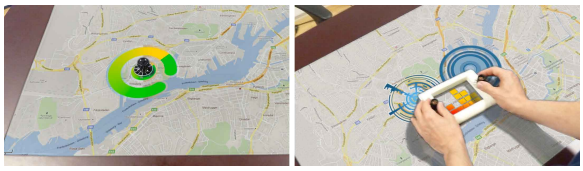


Figure 3: *Multi-state device on existing tabletop system showing tangible continuous parameter control for values such as time, radiation level, or population (left); selection frame with two-handed continuous control of radar visualizations or mode selectors (right). (Simulated images)*

2.1. Device tracking

Since a tabletop system is both a display and a multitouch input device at the same time, the complementary use of physical devices providing input may benefit certain types of applications [SR09]. Such devices may improve the fluidity and reduce the cognitive load of the user-system interaction [FB09]. Thus, the tabletop system must be able to distinguish between intended input from devices, and unintended input from other objects on the table [KF10]. Furthermore, tabletop systems must be able to detect multiple devices simultaneously when one or multiple users are interacting with the system.

2.2. Tracking in Tabletop Systems

In order to enable intuitive interaction with the content visible on the tabletop, devices other than the mouse and keyboard must be used. There is a class of devices that are easily identifiable by their inherent function known as 'physical icons' or phicons [IU97]. In this case, each device usually has a static association so that the tabletop system is able to detect its identifier (ID) in addition to its position. Once the device's ID is known to the system, the underlying functionality is also defined since the association cannot be changed. However, other tabletop devices might have a dynamic association that allows for simpler detection algorithms. In the latter case, the devices have a more general character, and so the intuitiveness is only guaranteed by the displayed content, i.e., the graphical user interface (GUI). The dynamic association is user-triggered and follows predefined steps. These steps may require some learning on the part of the user.

Tabletop systems must be able to detect the position of an interaction device and, in the case of phicons or other specialized input devices, their ID. While it is important in a global context for the position of the device to be displayed on the tabletop's surface, the ID is relevant for integrating a device's specialized functionality into a specific application. More degrees of freedom (DOF) than given by planar interaction become relevant. For instance, the z-coordinate may be used to distinguish between writing and pointing in pen-based interaction. Additionally, the tracking and detection system's latency should be below the user's perceptual threshold, otherwise user irritation may occur. During normal operation on a tabletop system, various objects may be placed on the surface which are not meant for interaction, but which could interfere with the system, e.g. by shadowing effects. Unlike a mouse, which is a relative pointing device that detects the travelling distance and orientation, all tracking systems for tabletop systems allow absolute pointing: the object is detected at precisely the place where the user puts the device.

2.3. Tracking Active Devices

Our work has been inspired by two contributions in the area of tangible tabletops using active TUIs: SmartFiducial [HK11] and MightyTrace [HKK08] [Hof11]. In SmartFiducial, active tangibles communicate with a host computer using wireless radio frequency (RF) transmission. Object positions are detected using a visual tracking system, thus adding a further level of complexity to the TUI design, as well as the host computer. Moreover, there is always a risk of unintended interference with RF communication. Hence, we deemed this approach to be unsuitable for our application. Inspired by the fast IR-sensors in MightyTrace, we decided to apply them as additional sensors to a camera-based system. In MightyTrace, a matrix of IR sensors is used to detect the position and state of TUIs. Each device is assigned a specific time frame during which its LED is turned on. Thus, each device can be detected unambiguously. For sending states, there are even more time slots for each device. For example, if a device has eight different states it then needs four time slots, one for detecting its position and three for sending its state bits (thus eight different states). However, the main restriction with such a time-multiplexed approach is that the system update rate is limited not only by the number of devices on the surface, but also by the number of states each device may have.

2.4. Dual Mode Tracking Method

Here, we introduce a new approach to improve multi-state IR tracking of tangibles on tabletops. Mainly, we aim to design low-latency active devices, that is, devices with more states without reducing the high refresh rate of the system. Moreover, we are interested in a cost-effective solution. We suggest a distinct way to combine a low-cost camera for position detection and a low-cost IR receiver for state detection of each device, in what we call a dual mode approach.

Considering the physical size of a table and the states required for tools such as sliders, rulers, and dials, we may simultaneously have five devices on the surface, each with 2048 possible states or adjustment levels. To meet this requirement, we use two different receivers: a camera capable of detecting positions, and an IR receiver capable of detecting the states of the devices. Concerning latency requirements, two terms are frequently employed: "update rate," which is the number of positions and states being updated per second, and "lag," which is the "response time" of the system to user input.

Much like in QualiTrack [HNK09], the devices and the camera are synchronized. Moreover, each device is assigned to a specific time slot. All devices emit IR light on each frame, except on their assigned time slot (Figure 4). Since the camera sees all the devices in every frame except one, the average update rate of the system equals f (M-1)/M, where f is camera frame rate (Hz) and M is the maximum number

of devices we want to have on the table (which is preconfigured and constant in the system). Camera frames are indexed in cycles of M frames. Since newly placed devices wait for up to one cycle to start IR transmission, the maximum setup delay equals M/f. Thus, using the dual mode method, the number of devices does not reduce the system's update rate, nor does it increase its lag. Only the setup delay will be negatively affected. We can now unambiguously identify each device and its position. Device tracking uses a blob-tracking algorithm [SHB*10]. While we assume that a device has one LED source only, instrumenting a device with two or more sources and combining their positions can give device orientation [Hof11].

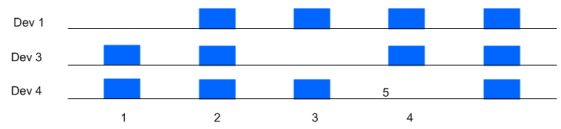


Figure 4: *Position information transfer example with M set to 5 and with devices 1, 3, and 4 (rows) present. A cycle of 5 frames (columns) is shown. Blue cells are IR flashes sent by the devices to be detected by the camera.*

Each device transmits its state information using its IR LED between two synchronization signals, i.e., the speed of transmitting the state information is significantly higher than the speed of the camera (Figure 5). This is feasible since the state information is read by a simple IR receiver and not by the camera. The interval between two consecutive camera frames is further divided into M sub-frames. Within each sub-frame, only the corresponding device sends its state information. The bit rate, R, of the sensor we employ can be up to 22 kbps. Hence, with the camera exposure time, e, and f and M as defined above, the maximum number of state bits per device equals: R ((1/f - e)/M). For example, with M set to 5, f at 60 Hz, and e at 10 ms, each device can transmit up to 29 bits of state information, allowing more than half a billion states per device.

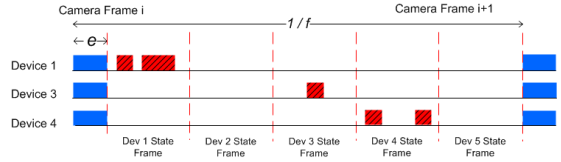


Figure 5: *State information transfer example with M set to 5 and with devices 1, 3, and 4 (rows) present. One interval between camera frames i and i+1 is shown. Blue cells are IR flashes to be detected by the camera; red cells are IR flashes to be detected by the IR receiver.*

2.5. Current status and future works

At an application level, we plan to further examine tasks requiring the use of multi-state tangible devices on tabletops. To this end, we plan to realize the slider, ruler, and dials accompanying the pen for use with our map-based tabletop application. Drawing on design principles and solutions from previous work [HKK08] [HNK09] [GSK06], this will call for engineering new TUIs tailored to this use. In a later phase, we foresee designing, running, and analyzing user studies to validate the usability and acceptance of the solutions.

At a tabletop device tracking level, we have evaluated the feasibility of the proposed method by implementing its essential subsystems. In particular, we implemented the IR receiver and changed the QualiTrack TUIs to send state information using our dual mode method. We also investigated whether the battery operated TUIs allow us to send signals powerful enough to be detected by the IR receiver, considering the distance between them. Our findings show that it is feasible to implement a tabletop using our new method. A next step in this project will be to implement a complete tangible tabletop using the method with the suggested slider, ruler, and dials.

2.6. Acknowledgements

We greatly thank Erik Tobin for improving the readability of this text.

References

[FB09] FJELD M., BARENDREGT W.: Epistemic action: A measure for cognitive support in tangible user interfaces? *Behavior Research Methods 41*, 3 (2009), 876–881. 3

[GSK06] GANSER C., STEINEMANN A., KUNZ A.: Infractables: Multi-user tracking system for interactive surfaces. In *Proc. of the IEEE conference on Virtual Reality (VR '06)* (2006), pp. 253–256. 1, 2, 4

[HK11] HOCHENBAUM J., KAPUR A.: Adding z-depth and pressure expressivity to tangible tabletop surfaces. In *Proceedings of the International Conference on New Interfaces for Musical Expression* (Oslo, Norway, 2011), Jensenius A. R., Tveit A., Godøy R. I., Overholt D., (Eds.), pp. 240–243. 3

[HKK08] HOFER R., KAPLAN P., KUNZ A.: Mighty trace: Multiuser technology on lcds. In *Proc. CHI '08* (2008), pp. 215–218. 3, 4

[HNK09] HOFER R., NESCHER T., KUNZ A.: Qualitrack: High-speed tui tracking for tabletop applications. In *Human-Computer Interactionâ INTERACT 2009*, Gross T., Gulliksen J., Kotz P., Oestreicher L., Palanque P., Prates R., Winckler M., (Eds.), vol. 5727 of *Lecture Notes in Computer Science*. Springer Berlin, Heidelberg, 2009, pp. 332–335. 3, 4

[Hof11] HOFER R.: *Tracking technologies for interactive tabletop surfaces*. PHD Thesis, ETH Zurich, 2011. 3, 4

[IMH05] IGARASHI T., MOSCOVICH T., HUGHES J.: As-rigid-as-possible shape manipulation. *ACM Trans. Graph. 24*, 3 (July 2005), 1134–1141. 2

[IU97] ISHII H., ULLMER B.: Tangible bits: Towards seamless interfaces between people, bits and atoms. In *Proc. CHI '97* (1997), pp. 234–241. 3

[KF10] KUNZ A., FJELD M.: From table-system to tabletop: Integrating technology into interactive surfaces. Human-Computer Interaction Series. 2010, pp. 53–72. 1, 3

[SHB*10] SCHOENING J., HOOK J., BARTINDALE T., SCHMIDT D., OLIVER P., ECHTLER F., MOTAMEDI N., BRANDL P., ZADOW U.: *Building Interactive Multi-touch Surfaces, Tabletops - Horizontal Interactive Displays*. Springer, 2010. 4

[SJG*06] SHAHROKNI A., JENARO J., GUSTAFSSON T., VINNBERG A., SANDSJO A., FJELD M.: One-dimensional force feedback slider: going from an analogue to a digital platform. In *Proc. NordiCHI '06* (2006), pp. 435–456. 2

[SR09] SHEN C., RYALL K. E. A.: Collaborative tabletop research and evaluation: Interfaces and interactions for direct-touch horizontal surfaces. In *Interactive artifacts and furniture supporting collaborative work and learning*, Dillenbourg P., Huang J., Cherubini M., (Eds.). Springer Science and Business Media, New York, Oslo, Norway, 2009, pp. 111–128. 3

[WWJ*06] WEISS M., WAGNER J., JANSEN Y., JENNINGS R., KHOSHABEH R., HOLLAN J., BORCHERS J.: Slap widgets: bridging the gap between virtual and physical controls on tabletops. In *Proc. CHI '09* (2006), pp. 481–490. 2