The Fez Spider

This is the main menù that appears on Fez Spider Display. The user can choose an action touching one of the four buttons displayed.

When the program starts, all the screens are created and initialized, but only the main one is displayed. Every screen is StackPanel composed of a series of other StackPanels. On the main screen's layout, for example, buttons are StackPanels with a particular Background image. Every button has two touch events (TouchDown and touchUp) in which we implemented their actions. TouchDown events are used only to show that buttons are pressed, changing their background images.

TouchUp events are used to start tipical action of the button. At program start up also other components are initialized like network connection.

The C# Server Application

The C# server allows users to display images and videos sended by Fez Spider by Ethernet network. The user can also send this files by email. The application consist of a Window Form.

The program starts launching two main threads (StartRecordingServer, StartUpdateServer) that manage the comunications with the Fez Spider. They launch two sockets on two different ports to receive the first, the video streaming in "real time" (1 frame every 1 second), the second the pictures. The video streaming is displayed in the panel on the left and also converted in avi format and saved in a local Video folder while the picture are converted to bytes and saved in the local Window DataBase connected to the application itself.

The Fez Spider

This is the main menù that appears on Fez Spider Display. The user can choose an action touching one of the four buttons displayed.

When the program starts, all the screens are created and initialized, but only the main one is displayed. Every screen is StackPanel composed of a series of other StackPanels. On the main screen's layout, for example, buttons are StackPanels with a particular Background image. Every button has two touch events (TouchDown and touchUp) in which we implemented their actions. TouchDown events are used only to show that buttons are pressed, changing their background images.

TouchUp events are used to start tipical action of the button. At program start up also other components are initialized like network connection.

The Photo Button

When clicked, the program enables the camera streaming and the user can take a picture tapping on the screen or clicking on the confirm button on the case.

To do this, the camera.BitmapStreamed event is used. Whenever a picture is captured by the camera, this event is triggered and the bitmap is saved in a global variable and is shown on the display.
If SD card is not inserted, nothing happens and a beautiful explanatory image is shown.

The program at startup register mount and unmount events on the SD card so that its possible to react quickly in every case is necessary to use it.

Save Picture

The system asks to user if he wants to save the picture on the internal SD card or not. The choose can be selected taping on the relative buttons showed on the screen. If the X image (or the back button) is pressed, the camera streaming is showed again. If the check image (or confirm button) is pressed the picture is saved and the system displays the main menù.

At this point, user can choose to save or to discard the current shown picture.
Tapping on check image, the program starts to control if SD card is inserted and is mounted; on positive outcome, the program begin to save picture on it. To prevent that SD card is removed suddenly, it is unmounted after image saving in order to assert modifications.
Tapping on X image the system returns to streaming camera screen so the user can make another photo.
Saving operation is done in a background thread with lower priority in order to avoid loss of graphic reactivity.

The Photo Button

When clicked, the program enables the camera streaming and the user can take a picture tapping on the screen or clicking on the confirm button on the case.

To do this, the camera.BitmapStreamed event is used. Whenever a picture is captured by the camera, this event is triggered and the bitmap is saved in a global variable and is shown on the display.
If SD card is not inserted, nothing happens and a beautiful explanatory image is shown.

The program at startup register mount and unmount events on the SD card so that its possible to react quickly in every case is necessary to use it.


Prezi

The Photo Gallery Button

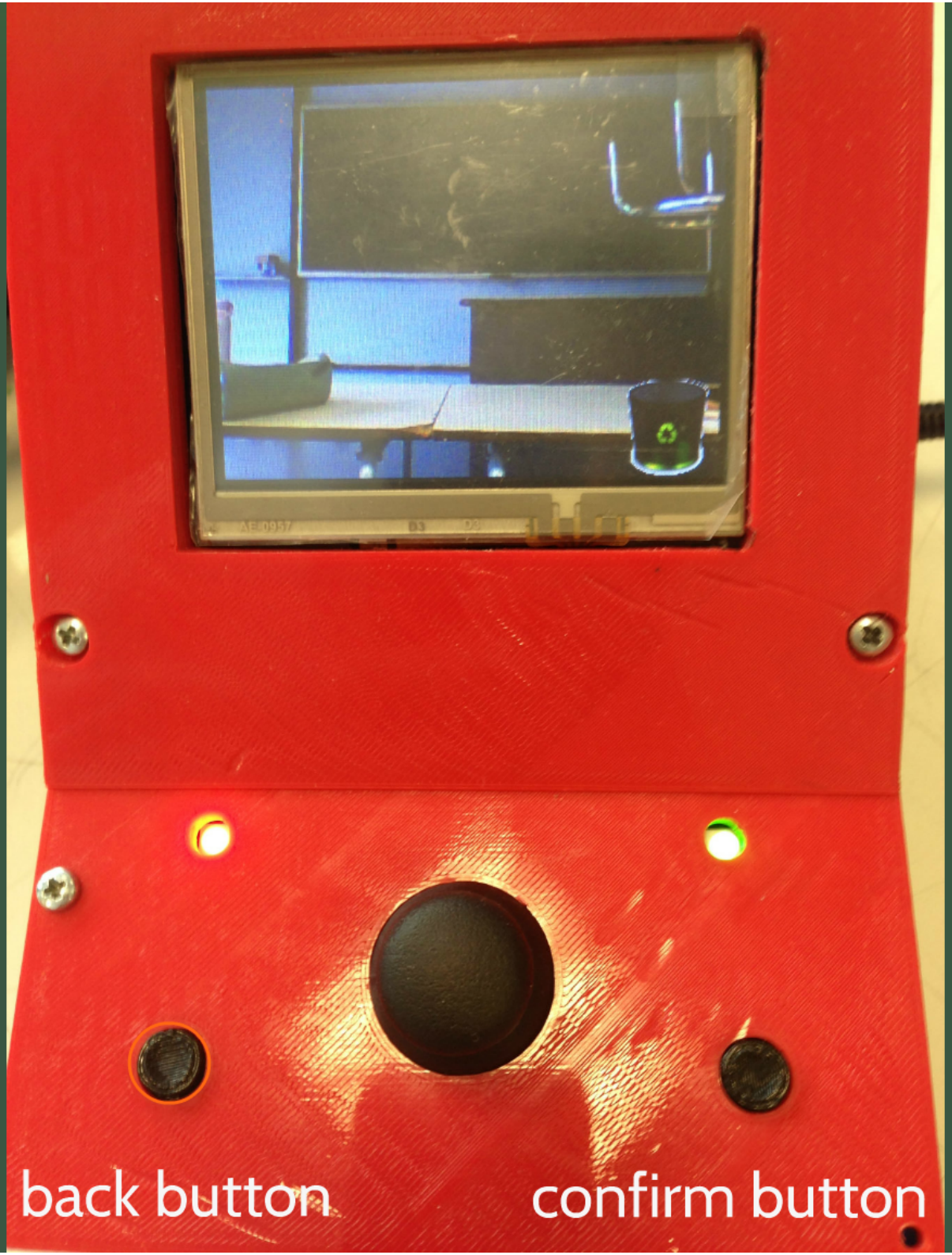
When user presses the photo gallery button the system showes, if there are saved photos, all the pictures stored in the SD card. The user can slide the pictures using the central joystick. He can also delete any pictures using the trash icon. If the user presses the back button the systems showes the main menù. Confirm button is unable in this state.

If no pictures are avalaible:

Another possibility is that the SD card is not inserted. In this case a explanatory image is shown and is impossible to access the photo gallery.

If at least a picture is avalaible:


No
image
available
AE-0957
D3
D3

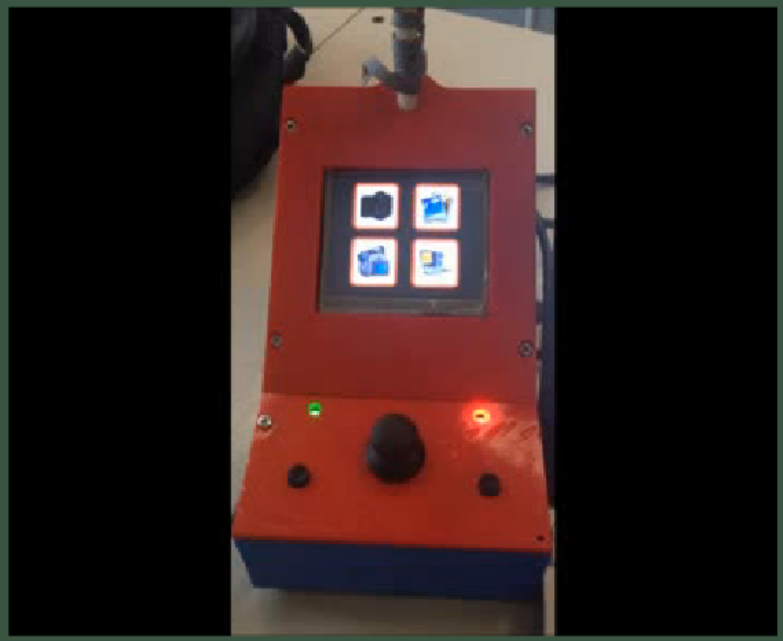



Prezi

The Video Streaming Button

When the user presses the video button the system starts the video streaming. The pictures are sent by ethernet to the server where are shown in "real-time" mode. When the user presses the back button the system turns off the streaming and goes back to the main menù then the server closes the connection and saves the video locally.

At this point a TCP connection and the camera stream are started. All is done using a global shared queue and a Wait/Notify mechanism. Whenever a picture is ready, it was saved in the queue by the camera BitmapStreamed handler, and notified by a static AutoResetEvent. The thread that opened the connection waits until a picture is available on the queue, sends it and waits again. The process ends when the user presses the back button.·

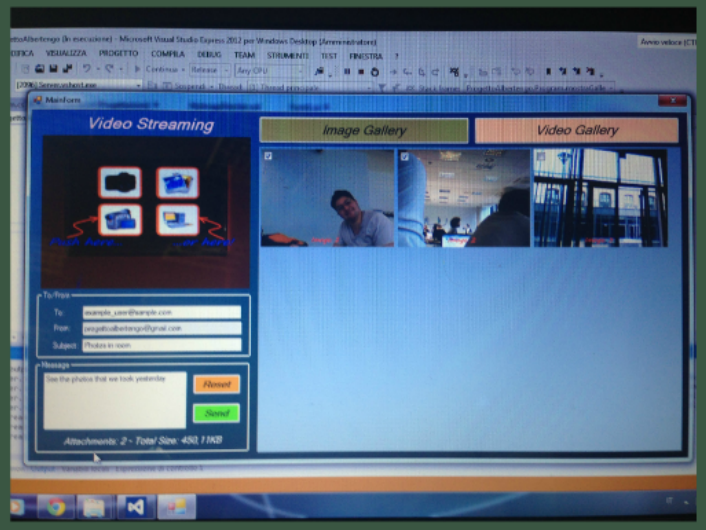



Prezi

The Transfer Button

When the user presses this button the system transfers all the stored pictures by ethernet to the server where he can display them pressing the ImageGallery. He can also open, delete and send by email both pictures and videos.

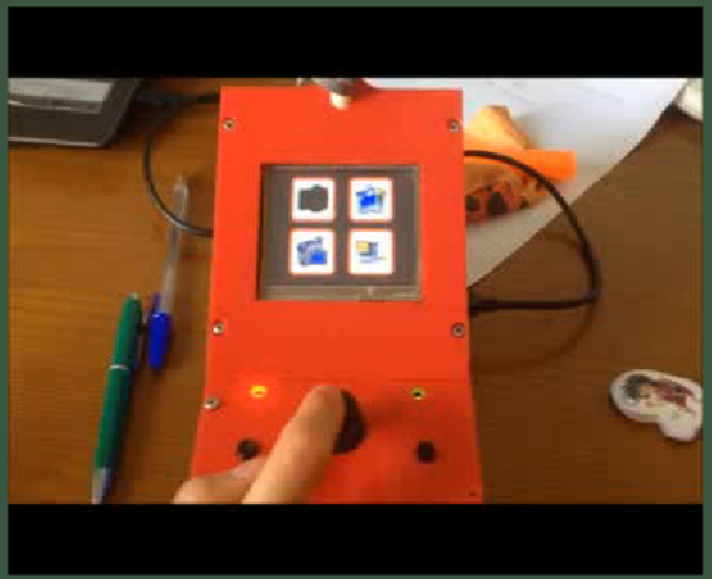
In this state the program starts a TCP connection to the Server, and sends all the images stored on SD card, loading them one by one. Even this feature works only if the SD card is mounted; in other case nothing happens. Images are converted in another a particular Bitmap format before sent, so that the Server can read them. (Bitmap objects are different in .NET and .NETMF)




Prezi

PACMAN !!!

If the user is boring he can press the joystick and the system will launch the PacMan game. If the user presses the back button the system returns to the main menù.



The Fez Spider

This is the main menù that appears on Fez Spider Display. The user can choose an action touching one of the four buttons displayed.

When the program starts, all the screens are created and initialized, but only the main one is displayed. Every screen is StackPanel composed of a series of other StackPanels. On the main screen's layout, for example, buttons are StackPanels with a particular Background image. Every button has two touch events (TouchDown and touchUp) in which we implemented their actions. TouchDown events are used only to show that buttons are pressed, changing their background images.

TouchUp events are used to start tipical action of the button. At program start up also other components are initialized like network connection.

The C# Server Application

The C# server allows users to display images and videos sended by Fez Spider by Ethernet network. The user can also send this files by email. The application consist of a Window Form.

The program starts launching two main threads (StartRecordingServer, StartUpdateServer) that manage the comunications with the Fez Spider. They launch two sockets on two different ports to receive the first, the video streaming in "real time" (1 frame every 1 second), the second the pictures. The video streaming is displayed in the panel on the left and also converted in avi format and saved in a local Video folder while the picture are converted to bytes and saved in the local Window DataBase connected to the application itself.

The Image Gallery Button

When the user clicks on this button the application showes,if they are present, the video sended by the Fez Spider in a Video Gallery visual tree. The application creates a PanelItem Object for each video and add it to the main layout. Every PanelItem has the image as background and several buttons and checkboxes that allow user to delete,select or open the picture or the video.

The button handler are defined in every panelItem. We create this type of Panels and the flowLayout because one problem encountered is to create this custom item in the C# listview and manage the handlers of the controlls in a easy way.

The C# Server Application

The C# server allows users to display images and videos sended by Fez Spider by Ethernet network. The user can also send this files by email. The application consist of a Window Form.

The program starts launching two main threads (StartRecordingServer, StartUpdateServer) that manage the comunications with the Fez Spider. They launch two sockets on two different ports to receive the first, the video streaming in "real time" (1 frame every 1 second), the second the pictures. The video streaming is displayed in the panel on the left and also converted in avi format and saved in a local Video folder while the picture are converted to bytes and saved in the local Window DataBase connected to the application itself.

The Video Gallery Button

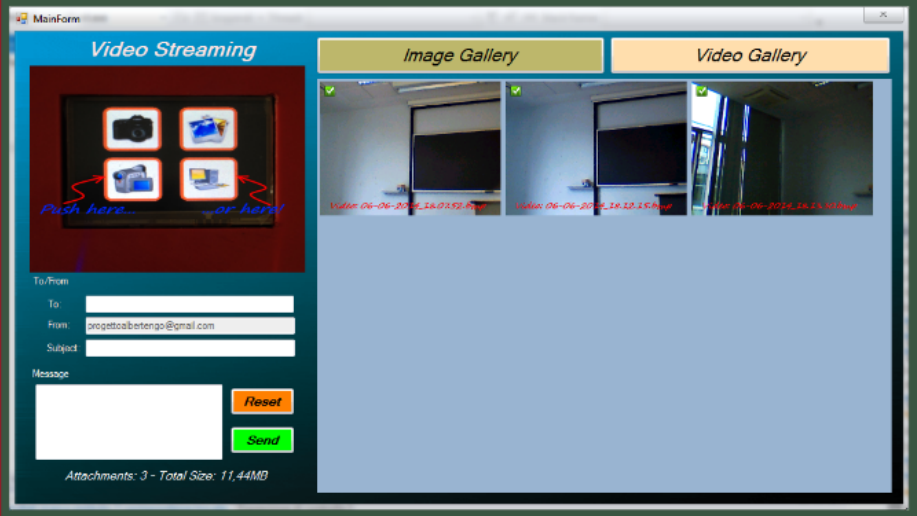
When the user clicks on it. The application showes the video stored in the Video folder and creates a PanelItem for every Video. The background image of the panelItem is the first frame of the video that the application saves in bmp format when the video is received. As for the Image Gallery every PanelItem has the controls to delete,select and open the video.

email. The application consist of a Window F

starts launching two main threads (StartRecordin

The Email Form

The user can selected both videos and images and send it as attachments by email (the attachments are not mandatory). The application when a checkbox is checked takes the dimension of the file and gives to user the total size of the attachements. The user can select a limited number of attachments (24MB) because the send became slow. When the uses presses the Send button , the click handler checks the fields and if they are not empty a thread starts. The thread create a local zip file named attachments.zip and add it to the Emailer Object. The Emailer class has two SendMessage methods: one with the attachments and one without. The email is send by a smtp Client. In our case using gmail smtp server.



A problem occurred is with the zip file. At the beginning we want to delete every time this file but it was used from the application so we decide to not delete it but to overwrite.

The Fez Spider

This is the main menù that appears on Fez Spider Display. The user can choose an action touching one of the four buttons displayed.

When the program starts, all the screens are created and initialized, but only the main one is displayed. Every screen is StackPanel composed of a series of other StackPanels. On the main screen's layout, for example, buttons are StackPanels with a particular Background image. Every button has two touch events (TouchDown and touchUp) in which we implemented their actions. TouchDown events are used only to show that buttons are pressed, changing their background images.

TouchUp events are used to start tipical action of the button. At program start up also other components are initialized like network connection.

The C# Server Application

The C# server allows users to display images and videos sended by Fez Spider by Ethernet network. The user can also send this files by email. The application consist of a Window Form.

The program starts launching two main threads (StartRecordingServer, StartUpdateServer) that manage the comunications with the Fez Spider. They launch two sockets on two different ports to receive the first, the video streaming in "real time" (1 frame every 1 second), the second the pictures. The video streaming is displayed in the panel on the left and also converted in avi format and saved in a local Video folder while the picture are converted to bytes and saved in the local Window DataBase connected to the application itself.