EDAF55 : Design Draft Surveillance Project

Due on Wednesday, November 8, 2017

Ola Johansson, Joachim Wedin, Hannes X, Erman X

Design 1

Camera - Server:

Threads:

takePicture:
has access to the camera monitor and uses mode to determine frequency of picture taking. Stores the images in a buffer in the monitor. Holds a variable (static?) which determines the wait time between images in idle mode.

sendPicture:
accesses picture buffer in monitor, wraps in package for sending over network, and sends. No logic concerning mode, sends asap.
updateMode:
waits for data packet from the client to set the mode in the cameraMonitor.
motionSensor:
a thread called from the client in order to check for motion, sends back motion data to client via http.
Monitors:
cameraMonitor:
methods for setMode, getImage and putImage. All these methods should be synchronized. the sendPicture thread might not be notified until buffer is full during movie mode to promote efficiency.
Client:
Threads:
MotionCtrl: Periodically checks the MotionSensor thread and receives motion data in return. Updates mode in the ClientMonitor.
SendMode: waits for update of mode in ClientMonitor and sends to UpdateMode thread in camera.
PictureReceiver: waits for new picture from sedPicture thread in camera. Stores pictures from both cameras in a common buffer, in the ClientMonitor.
Display: periodically updates the DisplayMonitor depending on the frequency of received pictures. Does not consider mode per se, instead uses time stamps of pictures to determine synchronous or asynchronous mode. There are two Display threads, one for each camera. Getting pictures from there respective buffers. While in synchronous mode sleeps the required time. While in asynchronous mode, does not sleep, sends to DisplayMonitor immediately. The display thread notifies the GUI that a new picture is available.
Input: sets mode regardless of motion in cameras.
Monitors:
ClientMonitor: holds methods for setting and getting mode, and a buffer for each camera storing pictures.
DisplayMonitor: holds references to two pictures, one for each camera. Updated by the two display threads. Read by the GUI thread. The methods for access are synchronized. These images are shown in the GUI.
GUI:
the GUI.
From picture to display - data flow:
picture is taken in the TakePicture thread. Picture is stored in the buffer of the CameraMonitor - notify SendPicture is notified and packs the picture with data (camera, timestamp, image size, image data) and sends via TCP PictureReceiver waits for picture from SendPicture. Gets image, saves in buffer in ClientMonitor. Display thread takes from the picture buffer ASAP. Depending on camera and frequency the pictures are displayed in different ways.
updating mode - data flow:

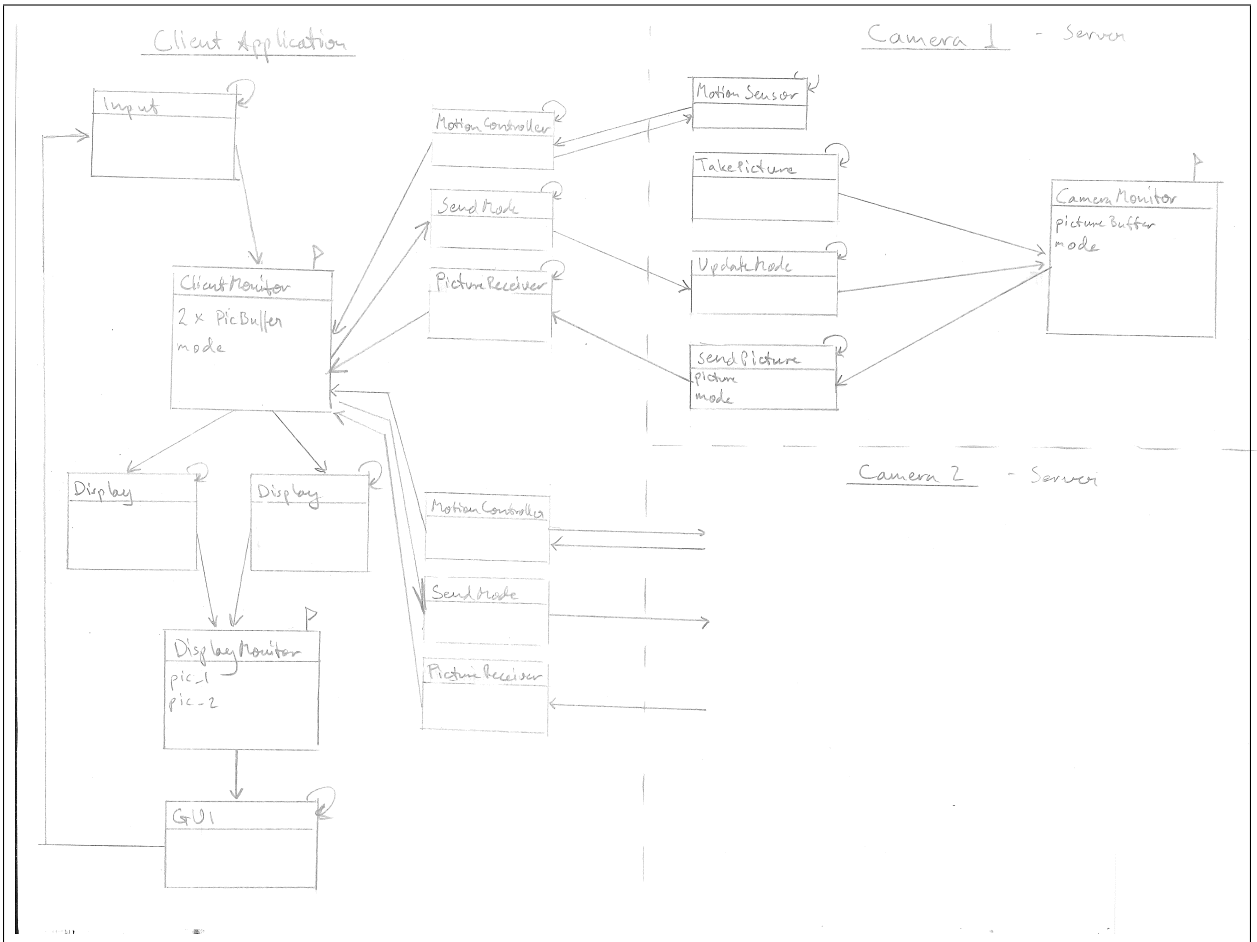
Design 1 continued on next page...

Periodically the MotionsCtrl thread requests info from MotionSensor thread via http, which informs the system of motion. MotionCtrl sets mode in ClientMonitor, notify. SendMode is notified and sends the mode update via TCP to the updateMode thread in camera. UpdateMode updates mode in the CameraMonitor.
Display
Packets:
Picture packet:
Time stamp no of bits: 32 size no of bits: 2Log(24 x [Resolution]) data no of bits: determined by size
mode packet:
a single bit representing the mode: 1 - movie 0 - idle
http communication:
existing protocol.

Design 2

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Design 2 continued on next page...



Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

Fusce mauris. Vestibulum luctus nibh at lectus. Sed bibendum, nulla a faucibus semper, leo velit ultricies tellus, ac venenatis arcu wisi vel nisl. Vestibulum diam. Aliquam pellentesque, augue quis sagittis posuere, turpis lacus congue quam, in hendrerit risus eros eget felis. Maecenas eget erat in sapien mattis porttitor. Vestibulum porttitor. Nulla facilisi. Sed a turpis eu lacus commodo facilisis. Morbi fringilla, wisi in dignissim interdum, justo lectus sagittis dui, et vehicula libero dui cursus dui. Mauris tempor ligula sed lacus. Duis cursus enim ut augue. Cras ac magna. Cras nulla. Nulla egestas. Curabitur a leo. Quisque egestas wisi eget nunc. Nam feugiat lacus vel est. Curabitur consectetuer.