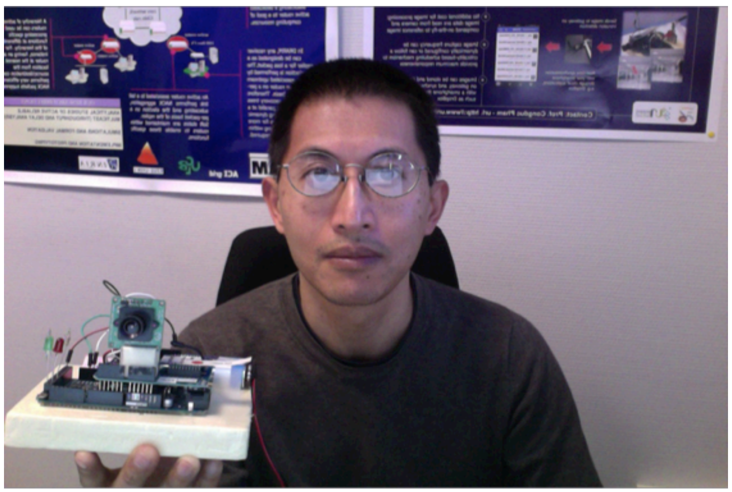
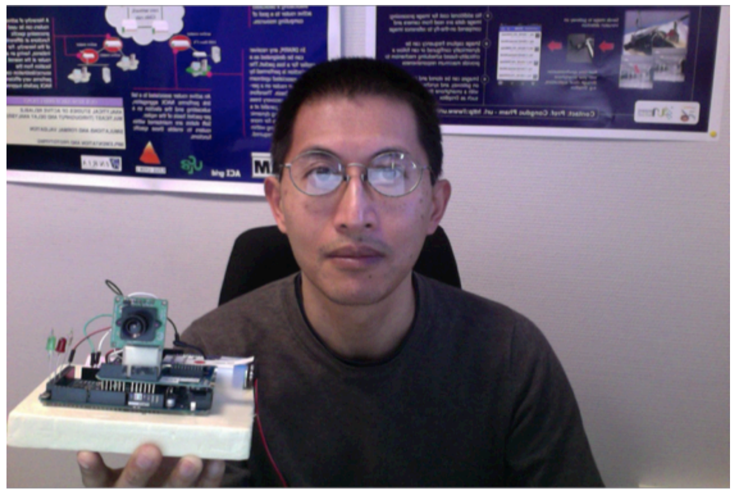
Low-Cost Wireless Image Sensor Networks for Visual Surveillance and Intrusion Detection



IEEE ICNSC'15
Taipei, Taiwan,
April 10th, 2015

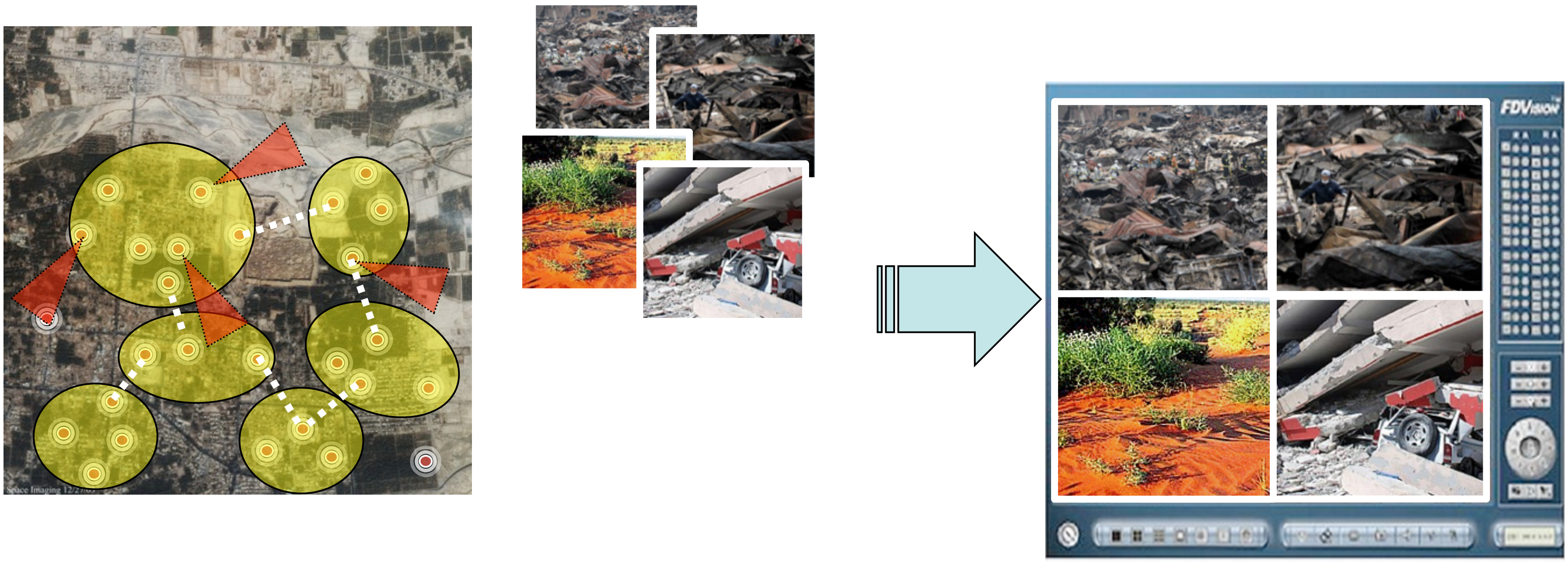
Prof. Congduc Pham
http://www.univ-pau.fr/~cpham
Université de Pau, France

Time for presentation



is about 20 minutes

Visual data for situation-awareness



Collect data to improve the responsiveness of rescue operations

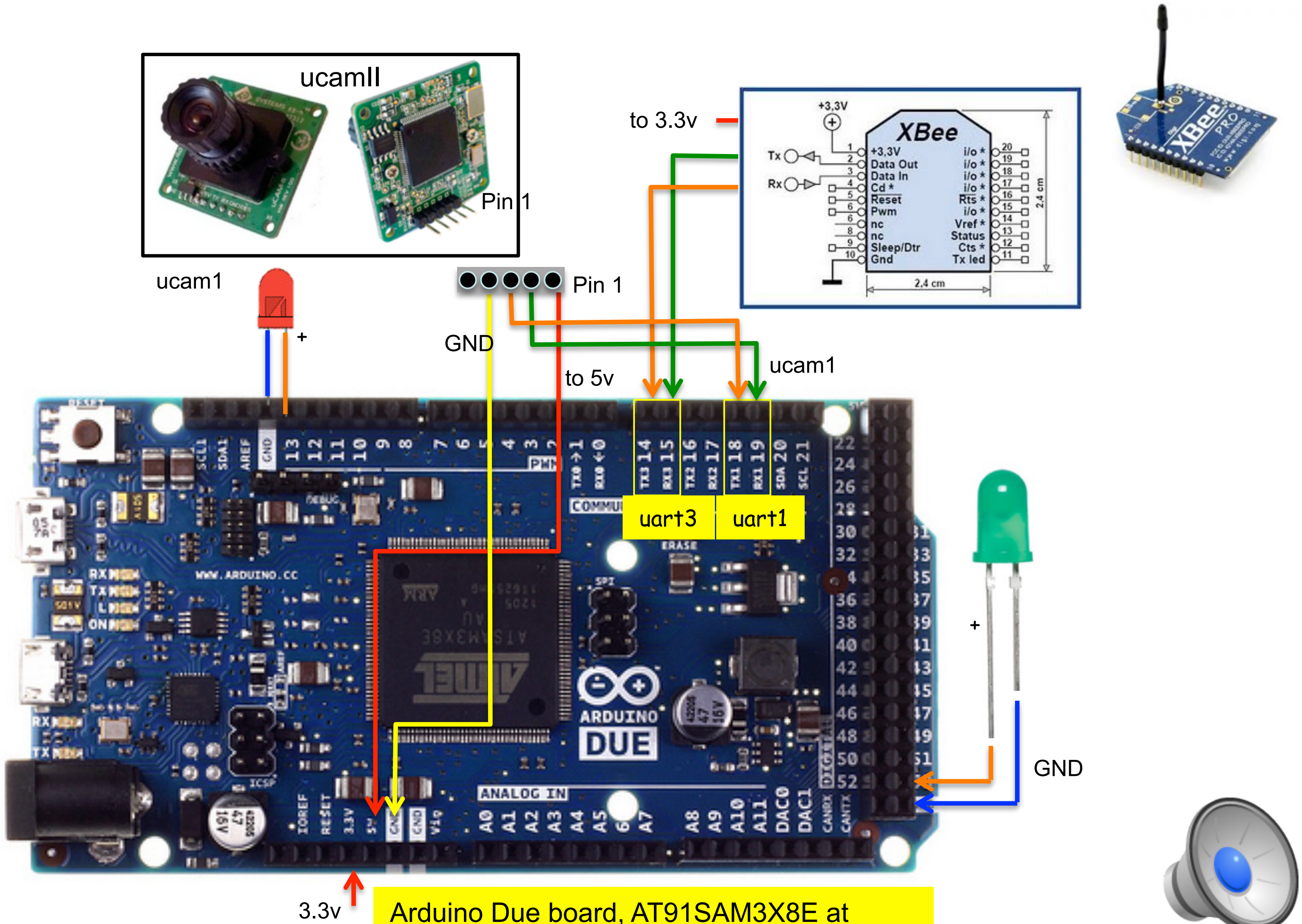
Visual/image sensors

- Cyclops
- SeedEyes
- CMUcam4 & CMUcam5(PIXY)
- iMote2/IMB400…
- Mostly based on ad-hoc development of the visual part (dedicated camera circuit or dedicated µC for image acquisition/processing)
- Image encoding mechanism rarely adapted to low-resource platform (memory, radio,…)
- Can hardly run out-of-the-box for surveillance applications

Motivations & Objectives

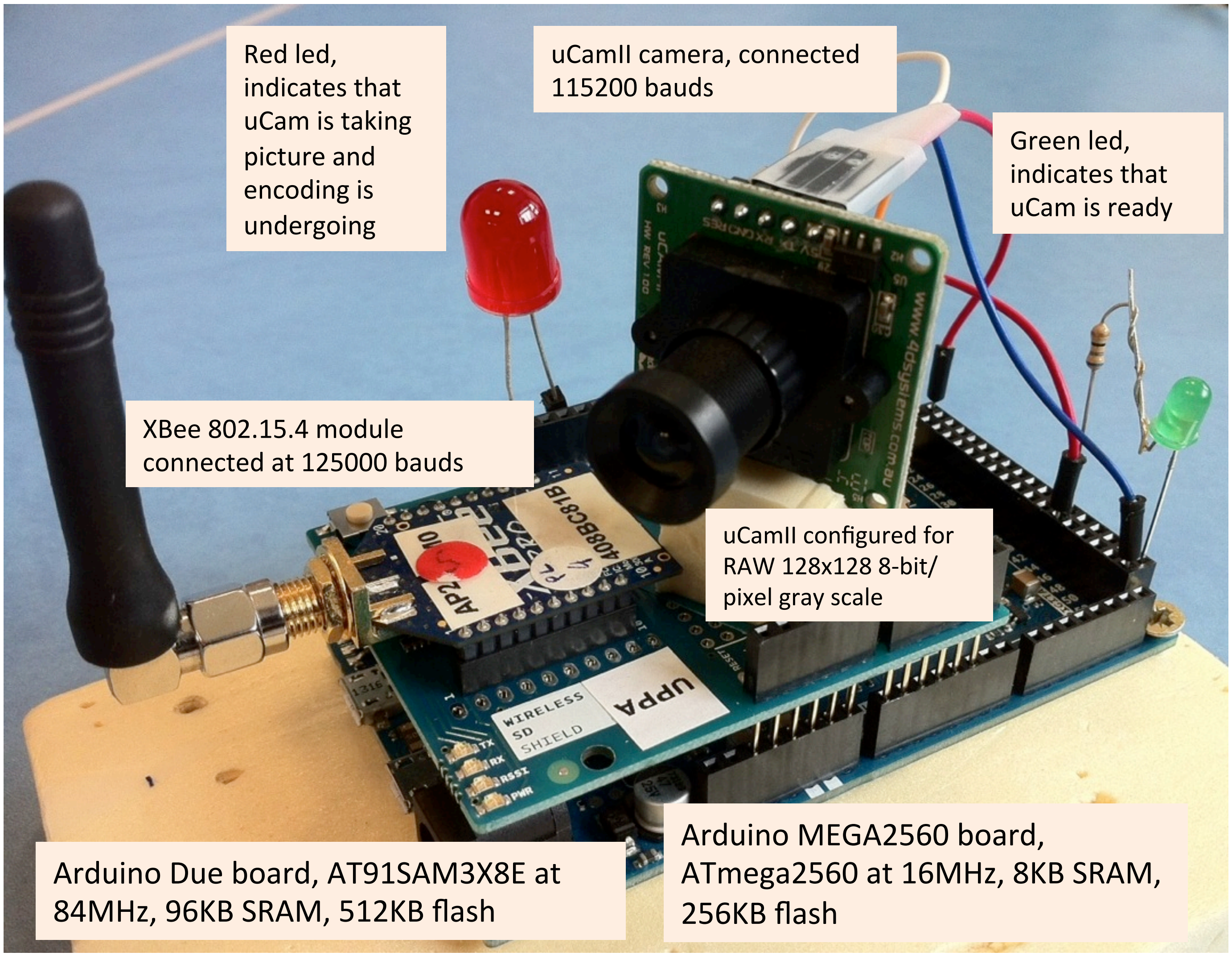
- Offer an off-the-shelf solution so that anybody can reproduce the hardware and software components
  - Arduino-based solution for maximum flexibility and simplicity in programming and design;
  - Simple, affordable external camera to get raw image data
- Integrate and apply fast and efficient compression scheme with the host μC (no additional nor dedicated μC)
  - Small size image
  - packet loss-tolerant bit stream

Arduino + uCamII
128x128 8bpp raw images



Arduino Due board, AT91SAM3X8E at 84MHz, 96KB SRAM, 512KB flash

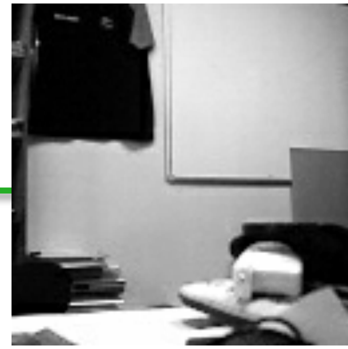
Our image sensor



ADJUSTABLE IMAGE QUALITY FACTOR Q

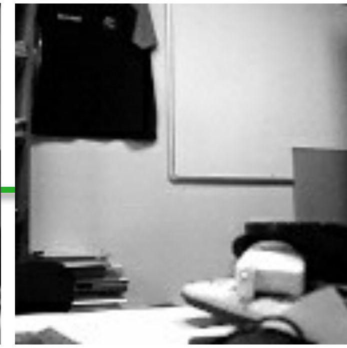
raw 16384b

Q=100; 9768b (1.67)



PSNR=51.344

Q=90; 5125b (3.2)



PSNR=29.414

Q=80; 3729b (4.4)

PSNR=28.866

Q=70; 2957b (5.5)

PSNR=28.477

Q=60; 2552b (6.4)

PSNR=28.024

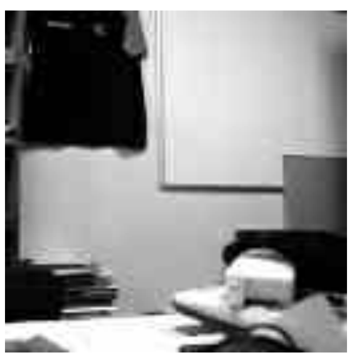
Q=50; 2265b (7.2)

PSNR=27.912

Q=40; 2024b (8.1)

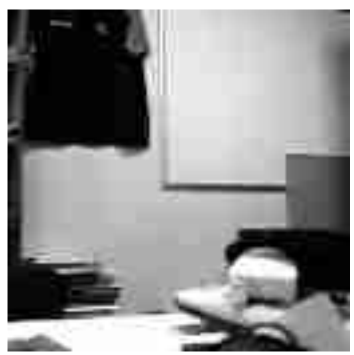
PSNR=27.423

Q=30; 1735b (9.5)



PSNR=26.933

Q=20; 1366b (12)



PSNR=26.038

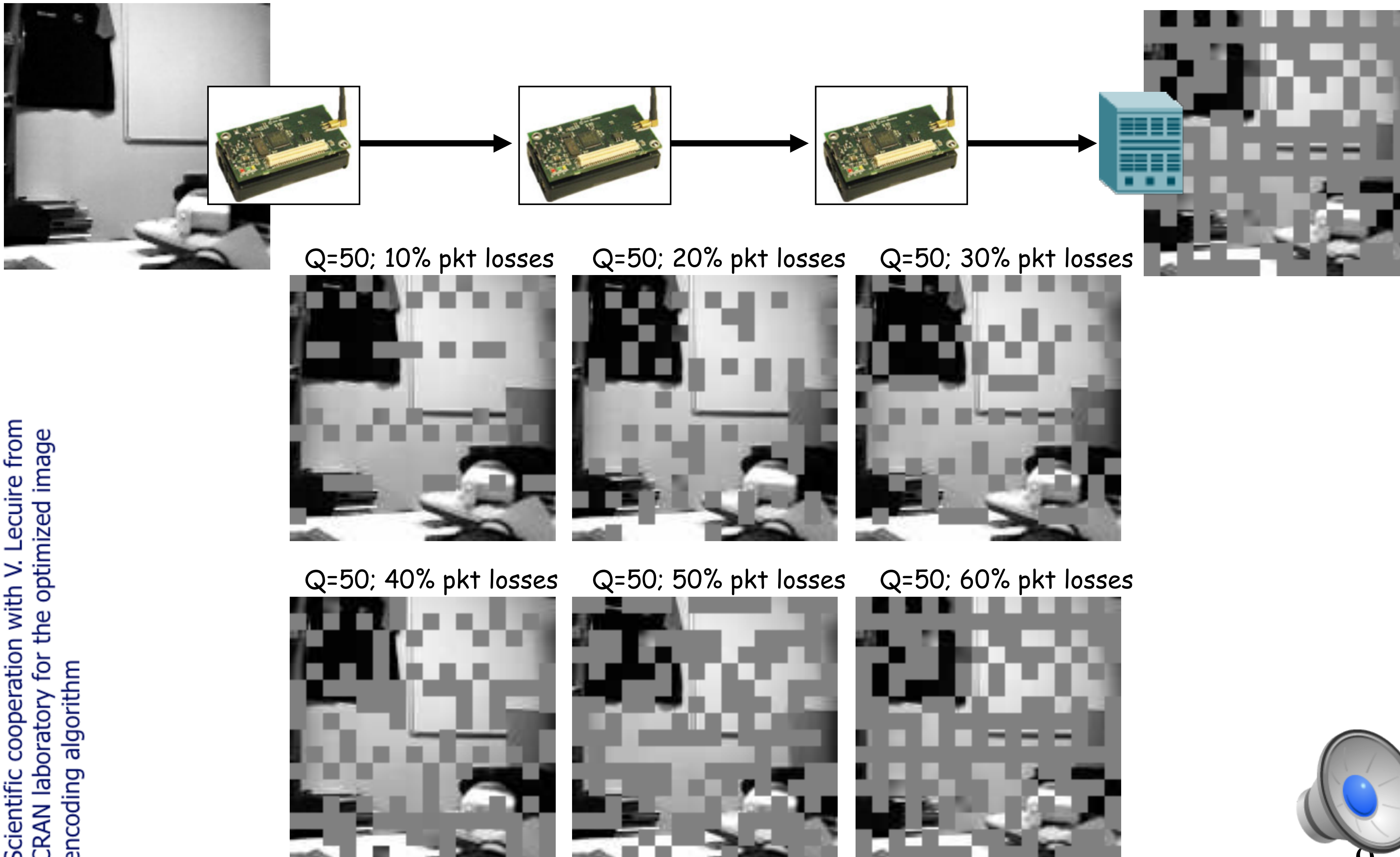
Q=10; 911b (18)

PSNR=25.283

Q=5; 576b (28.44)

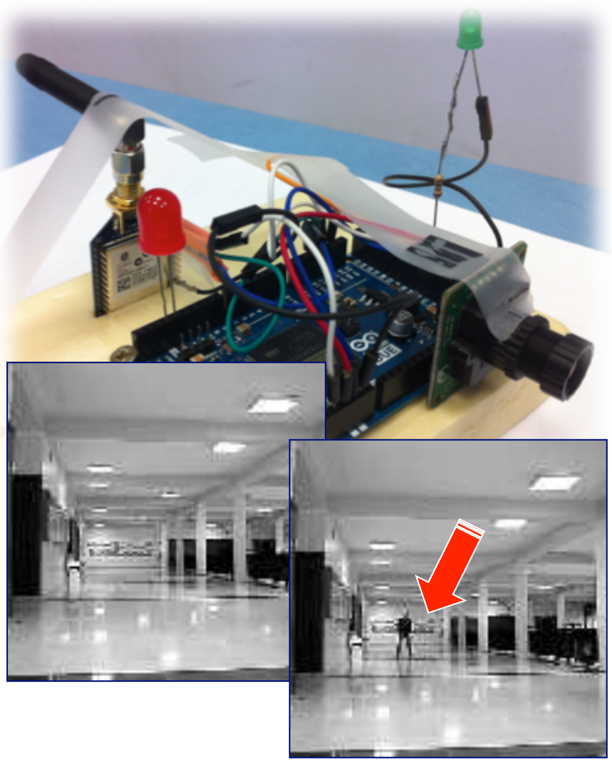
PSNR=23.507

PACKET LOSS-TOLERANT BIT STREAM, ANY RECEPTION ORDER



Scientific cooperation with V. Lecuire from CRAN laboratory for the optimized image encoding algorithm

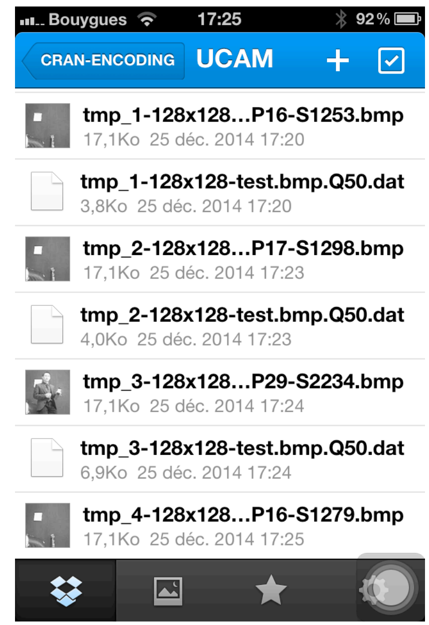
Image change detection for Intrusion detection



Very lightweight « simple-differencing » method, takes into account modification in image luminosity

Sends image to gateway on intrusion detection

Real-time synchronization with your smartphone through cloud applications, e.g. DropBox



Very low-memory platforms

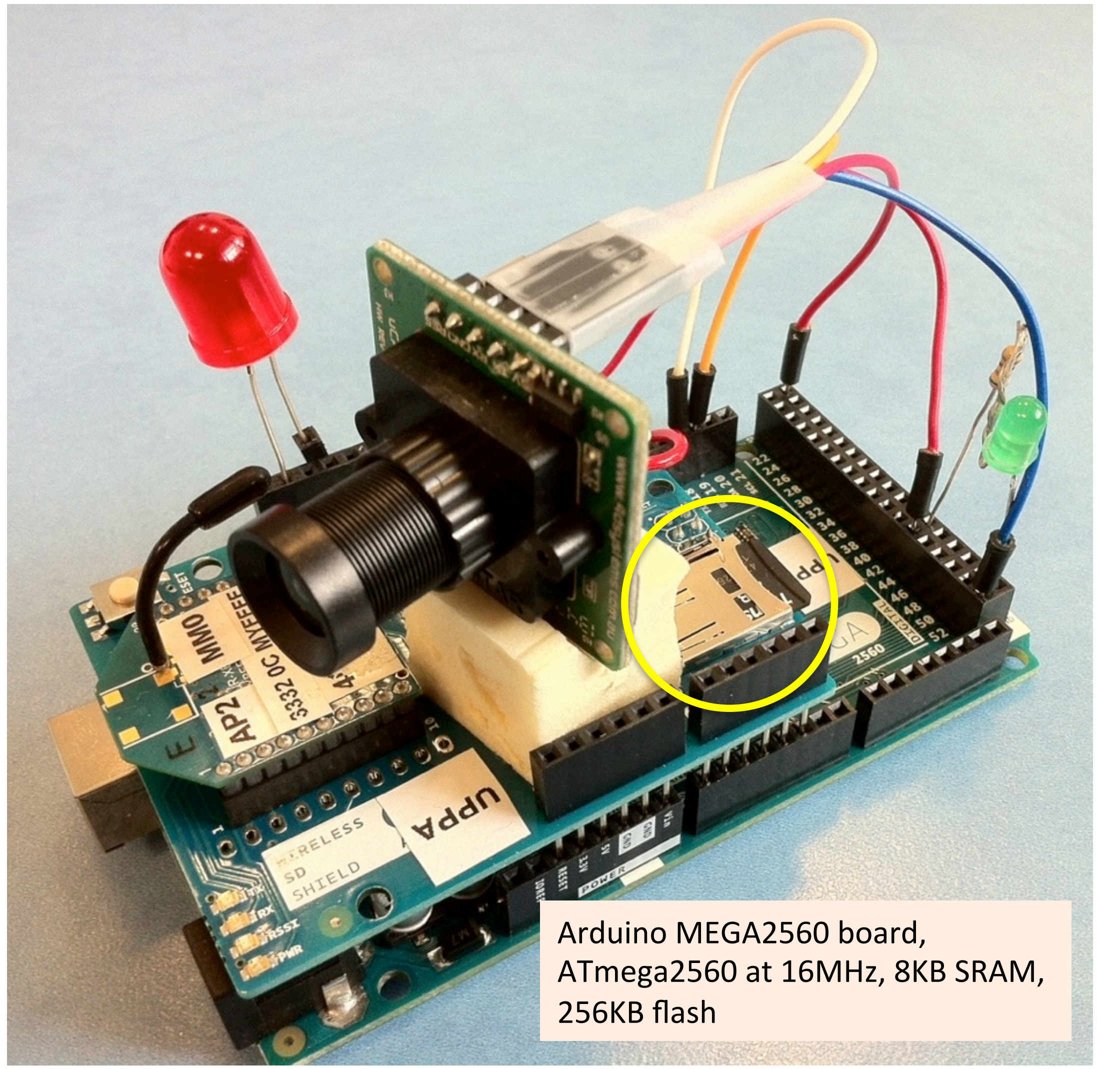
Arduino MEGA2560 at 16MHz, 8KB SRAM

Only 2KB SRAM available at runtime

Modified encoding algorithm to avoid having all the raw image in SRAM: encoding, packetization and transmission in a row per image packet

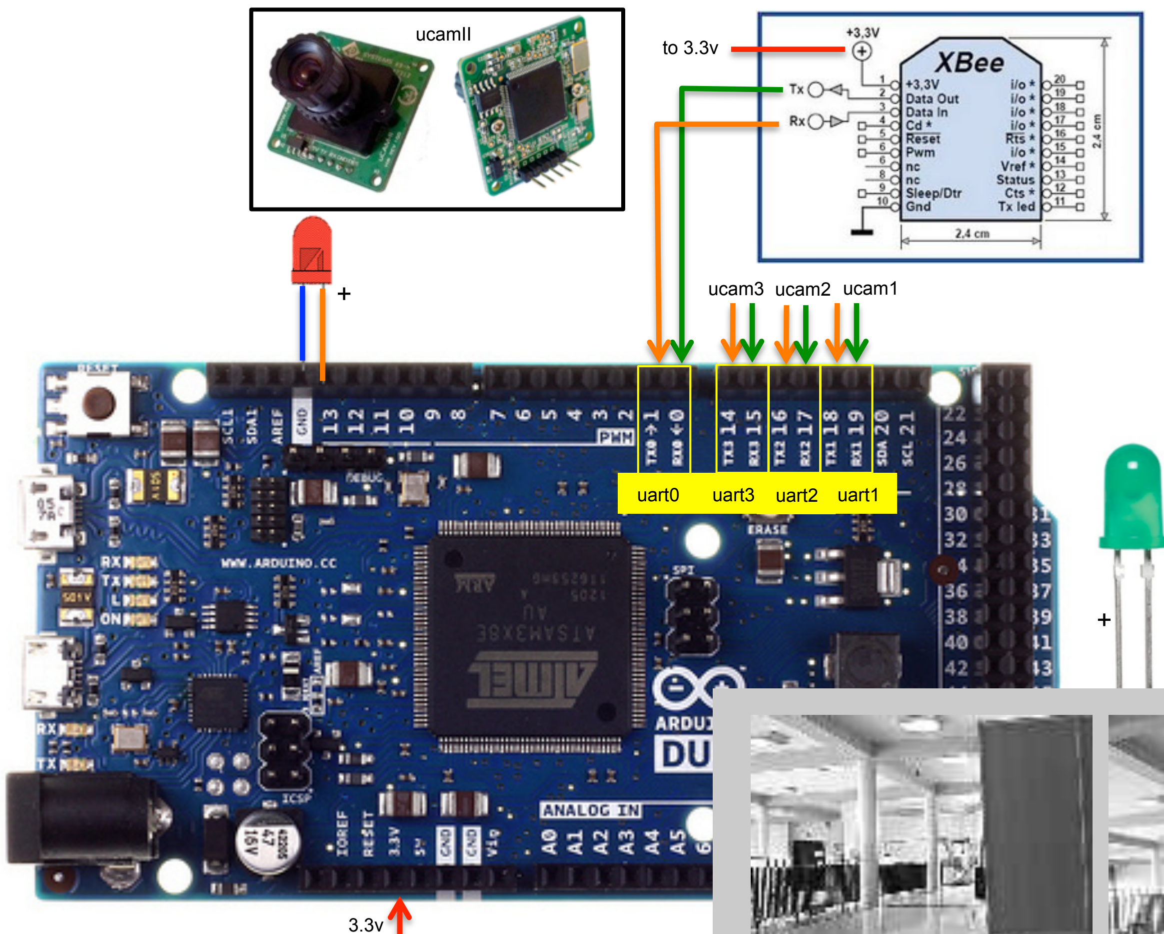
Reference image and current raw stored in an SD card

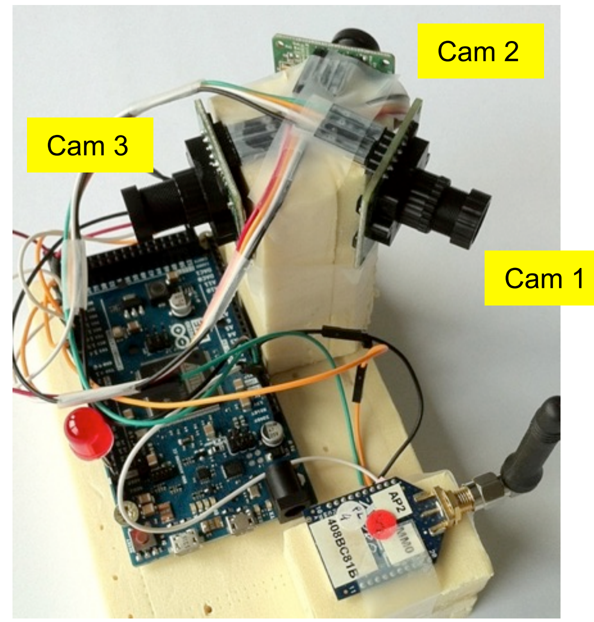
Encoding and packetization will read image blocks from SD card

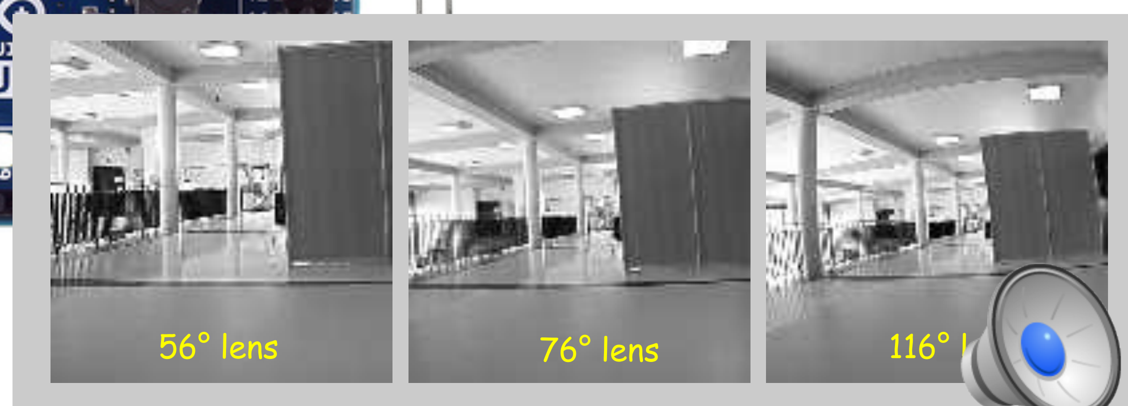


Arduino MEGA2560 board, ATmega2560 at 16MHz, 8KB SRAM, 256KB flash

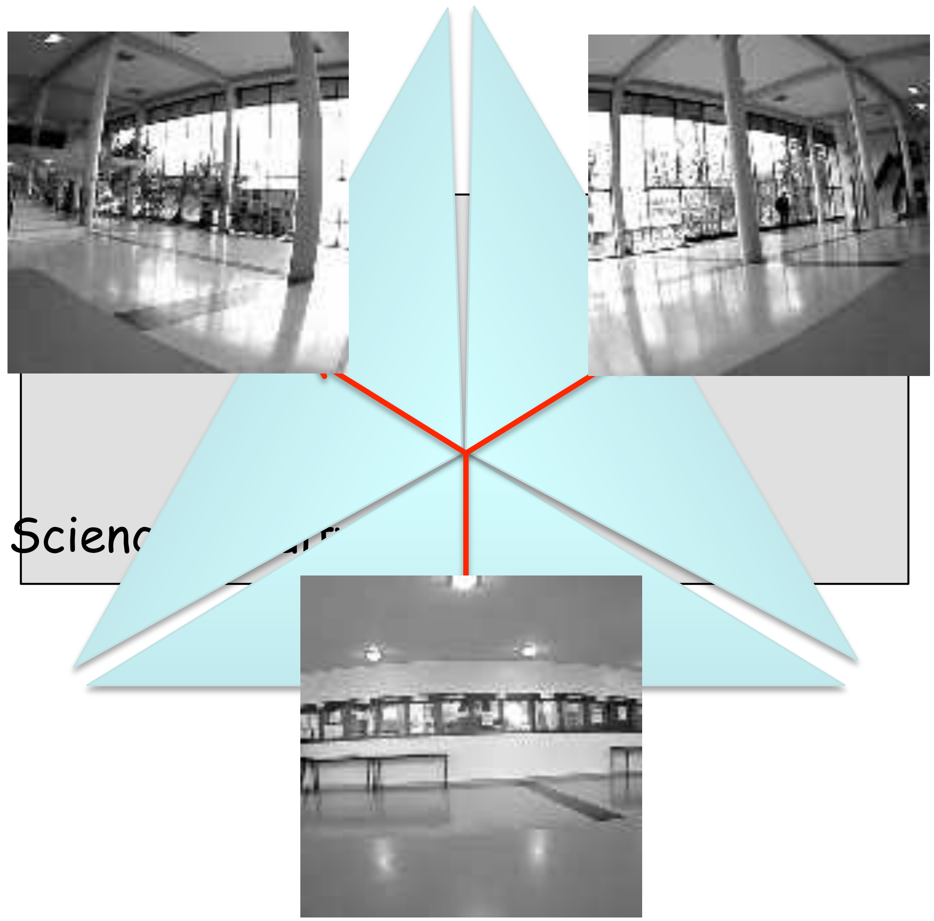
Multi-camera system

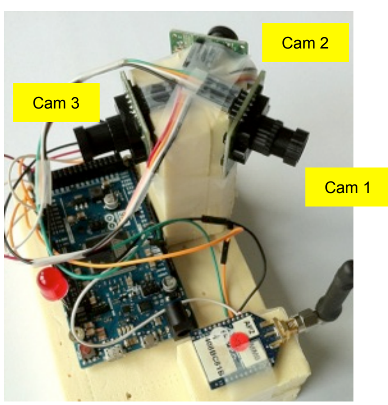


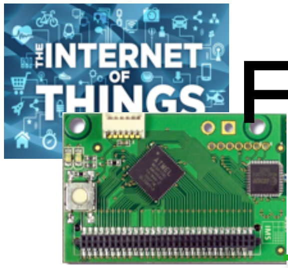




Low-cost omnidirectional visual sensing

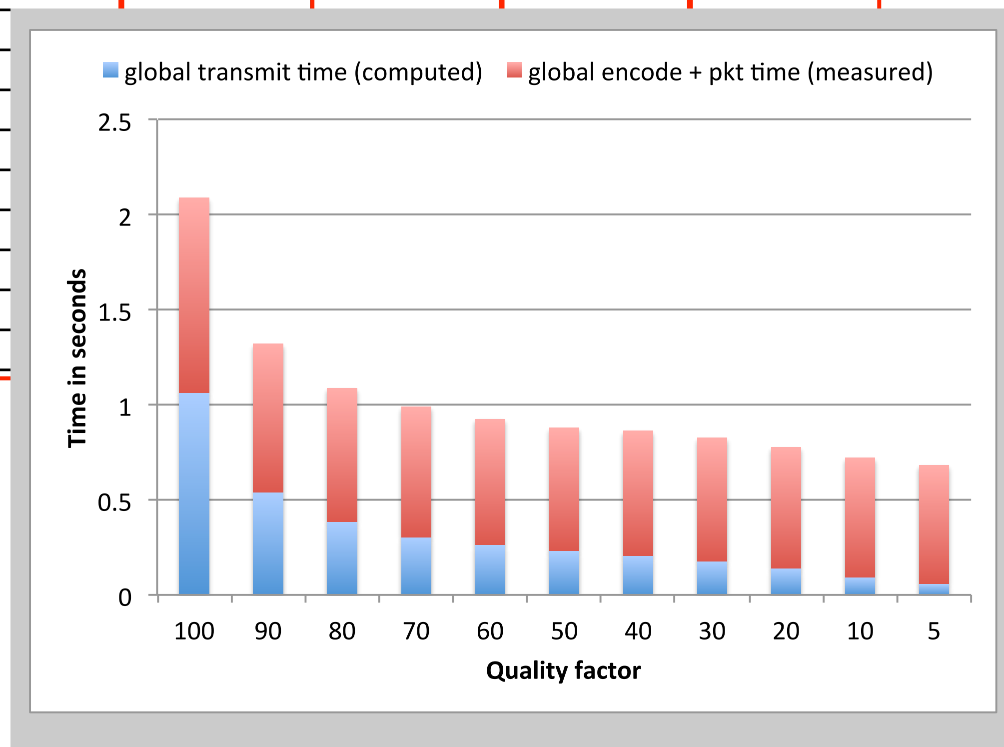






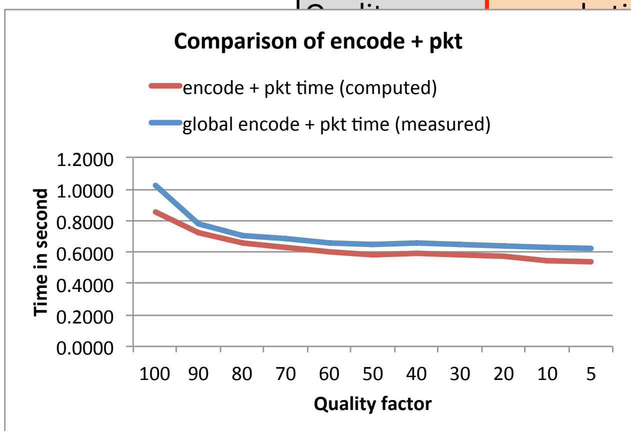
Performance measures Arduino Due

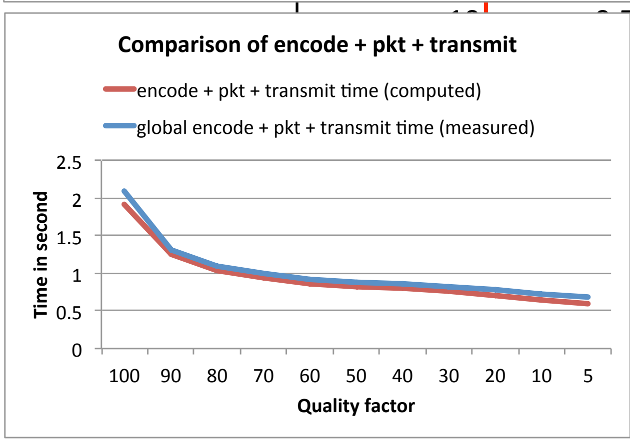
		N	R	A	B = D - A`	C = B / N	D	E = D / N
Quality Factor Q	size in bytes (MSS=90)	Number of packets	time to read data from ucam	global encode + pkt time (measured)	global transmit time (computed)	transmit time/pkt (computed)	global encode + pkt + transmit time (measured)	encode+transmit time/pkt (in ms)
100	9768	158	1.512	1.027	1.064	0.0067	2.091	13.2342
90	5125	70	1.512	0.782	0.539	0.0077	1.321	18.8714
80	3729	48	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
70	2957	37	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
60	2552	32	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
50	2265	28	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
40	2024	25	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
30	1735	21	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
20	1366	17	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
10	911	11	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
5	576	7	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

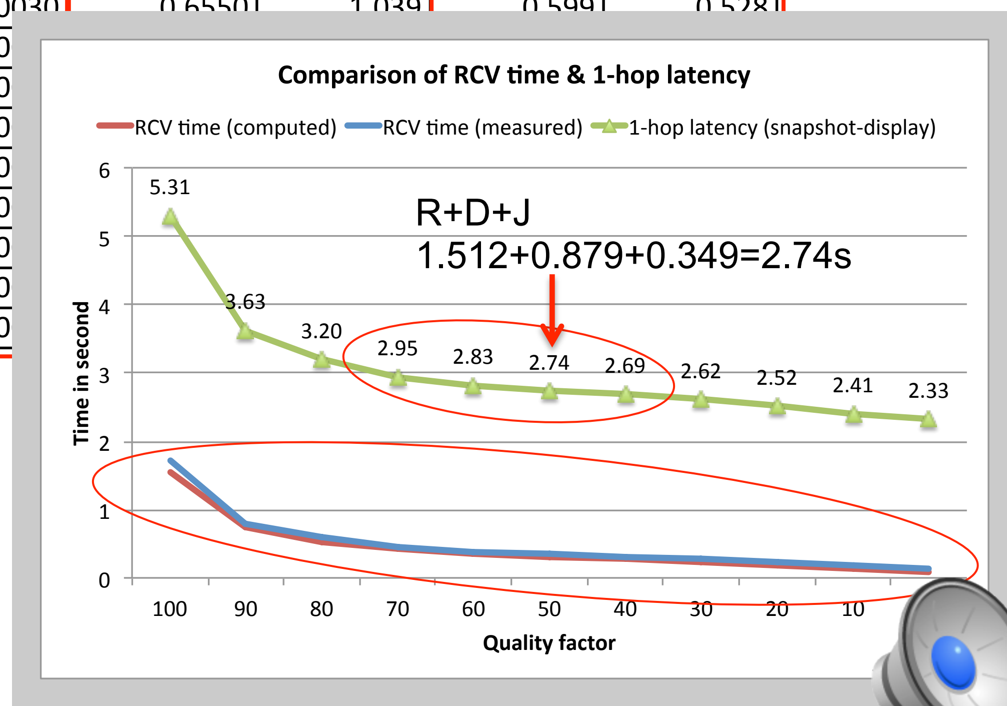


Detailed measures

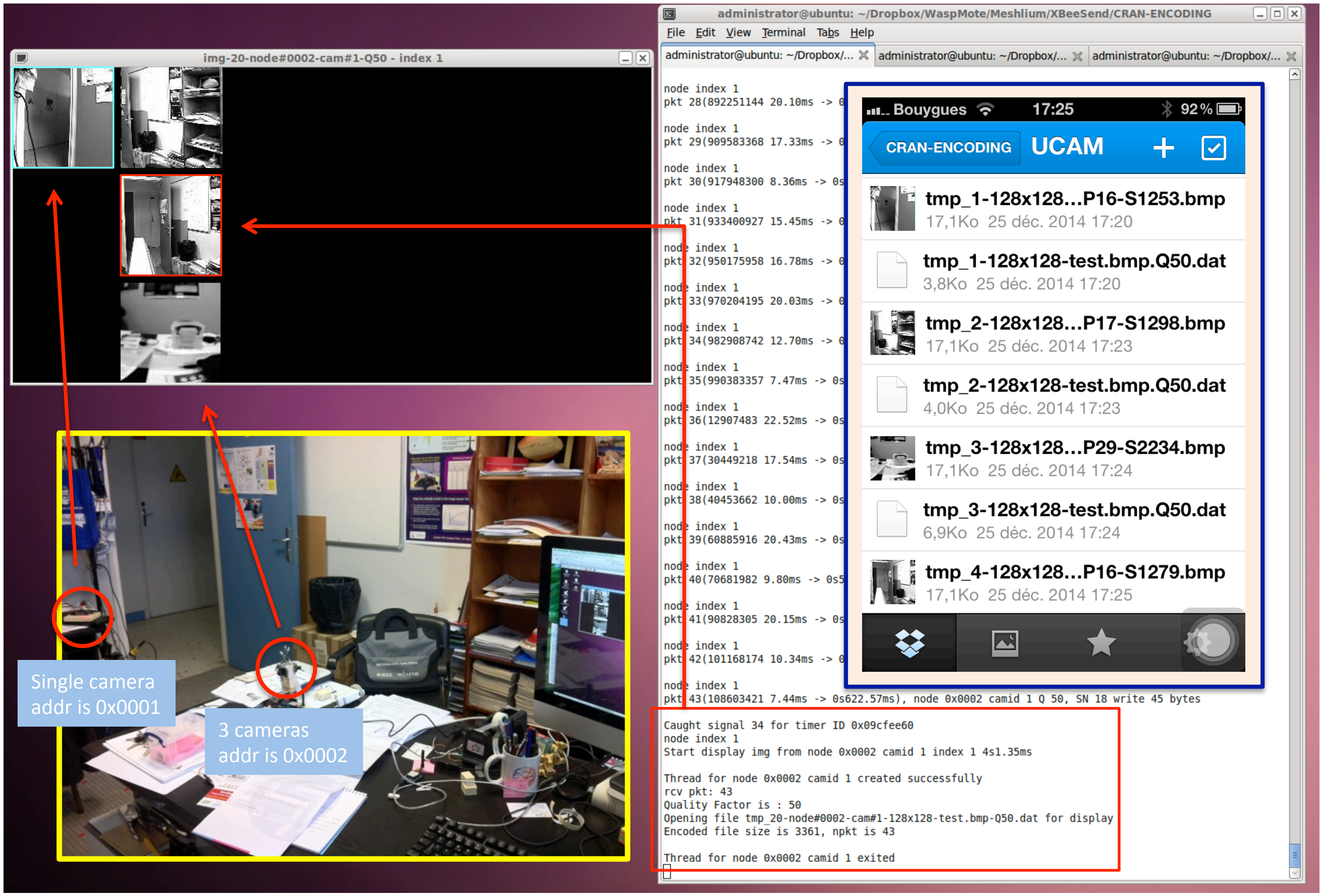
	F	G	H = F + G * N	I = H + C * N	J	K = (C + G)*N
Qualit…	…kt time …ed)	packetization (pkt) time (measured & rounded)	encode + pkt time (computed)	encode + pkt + transmit time (computed)	RCV time (measured)	RCV time (computed)
	…79	0.0030	0.8530	1.917	1.708	1.538
	…12	0.0030	0.7220	1.261	0.799	0.749
	…11	0.0030	0.6550	1.039	0.599	0.528
	…19	0.0…				
	…09	0.0…				
	…00	0.0…				
	…16	0.0…				
	…16	0.0…				
	…18	0.0…				
	…16	0.0…				
	…18	0.0…				







Out-of-the-box surveillance



Using realistic parameters in simulation

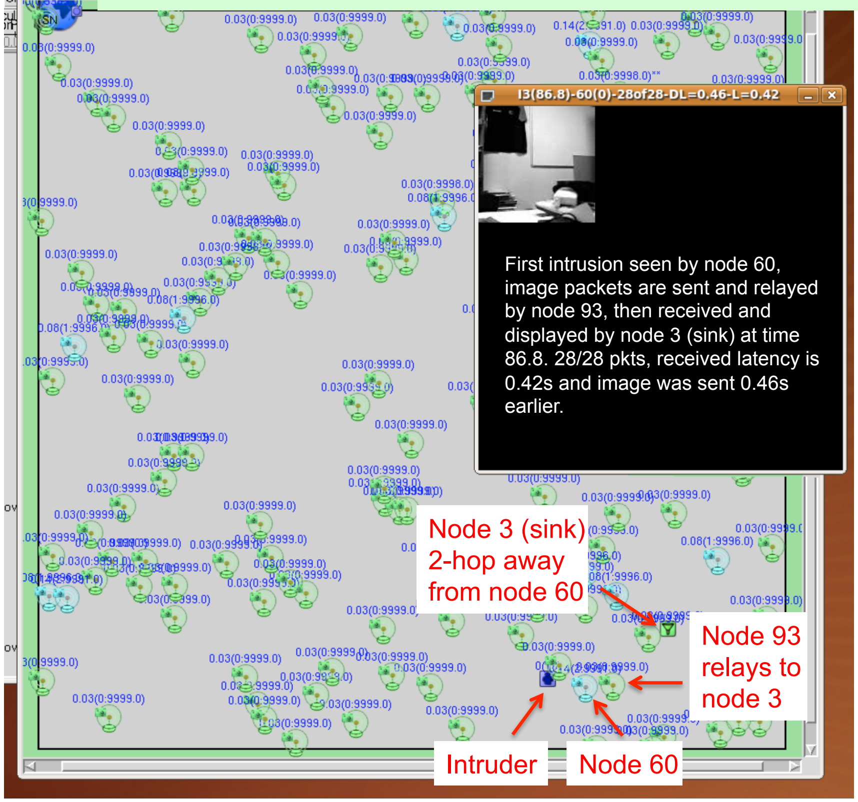
Q=50; 2265b

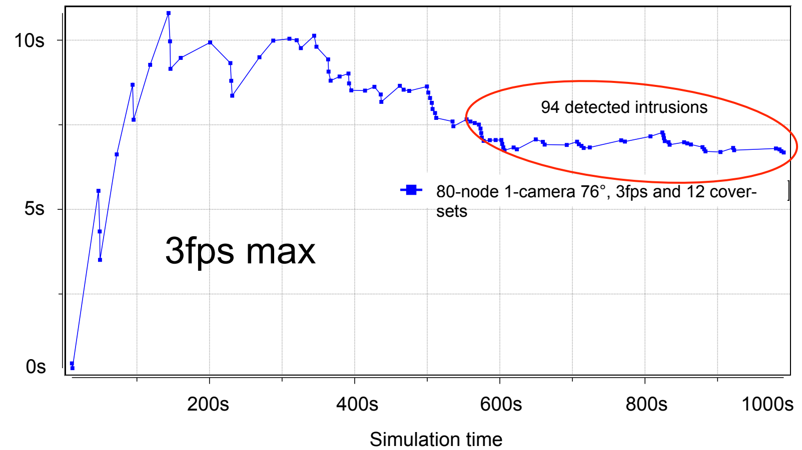
- 128x128 8bpp encoded image
- Quality Factor of 50: encoded image of 2265 bytes in 28 packets
- We need 200ms to configure the camera. Time before image data can be processed is 1.512s
- So the maximum frame capture rate is 0.58fps
- Camera angle of view could be 56°, 76° or 116°
- Depth of view of 25m
- Packet overhead at the image source is 11ms (8ms for transmission and 3ms for packetization)
- Packet relaying time is 16ms (based on measures of MicaZ platform)

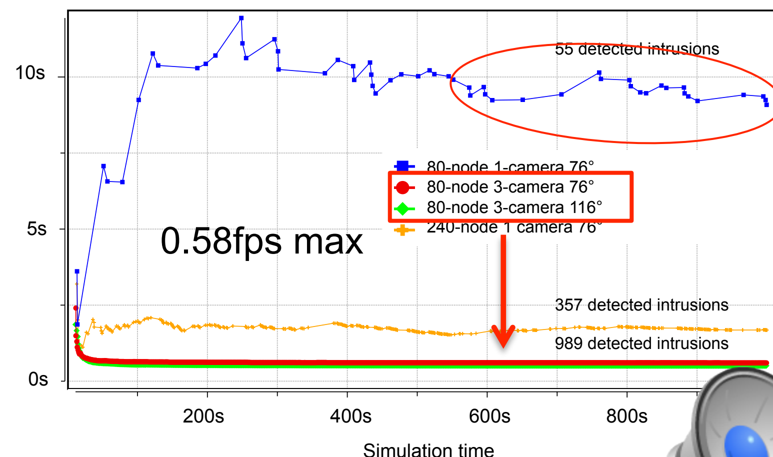
Simulation scenario

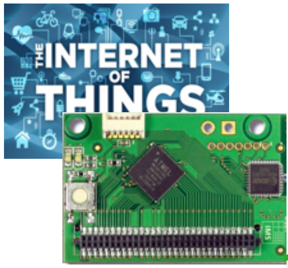
The simulation model is used to study performance of large-scale intrusion detection system

Using real measures for image processing tasks and packet transmission overheads produces very accurate simulation results that are consistent to what have been found in real multi-hop experimentations





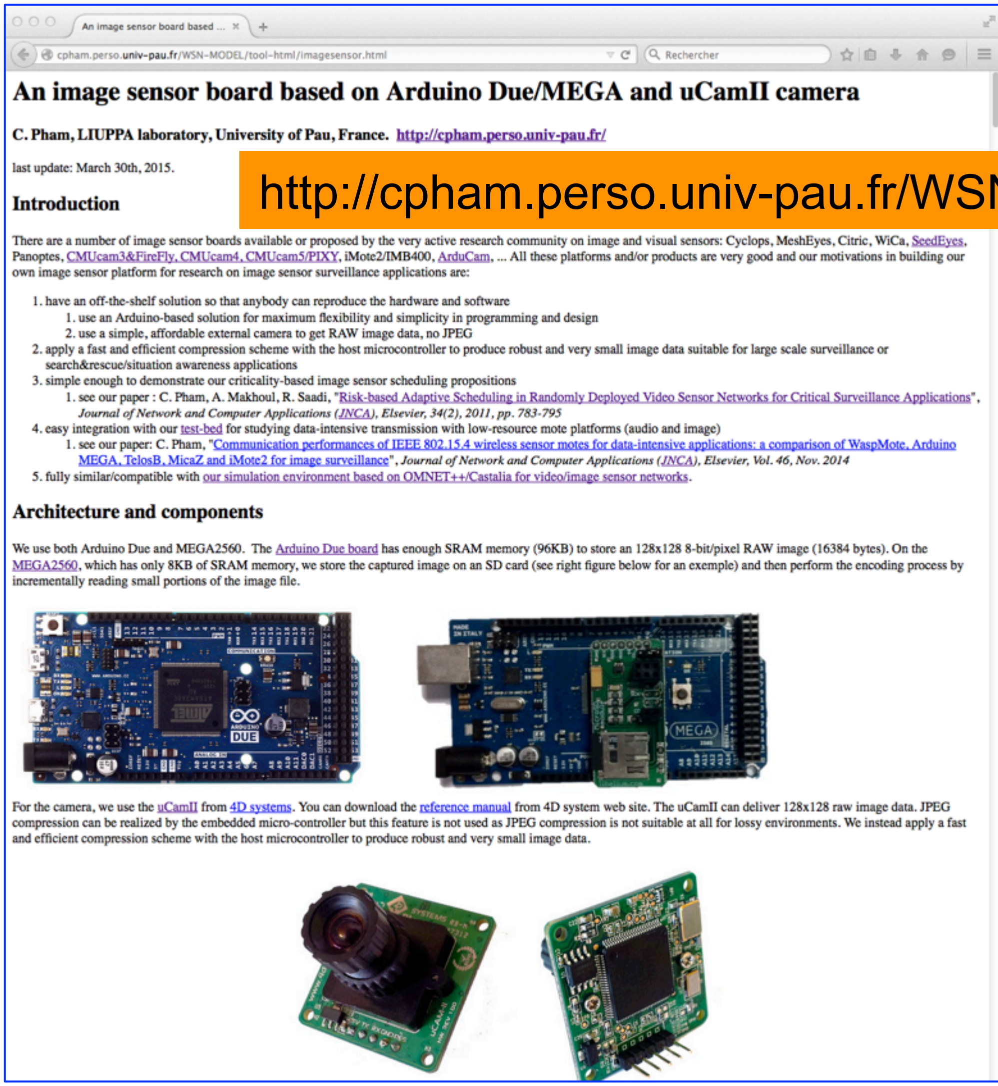




Conclusions

- Low-cost image sensor from off-the-shelves components with fast and packet loss-tolerant encoding
- Can run out-of-the box to perform surveillance tasks based on image change detection
- The image latency can be less than 2.3s using medium quality image
- At 1-hop, the receive & display latency can be less than 2.8s using medium quality image
- Detailed performance measures are used to produce more accurate large-scale simulation models

Web page


An image sensor board based on Arduino Due/MEGA and uCamII camera

C. Pham, LIUPPA laboratory, University of Pau, France. http://cpham.perso.univ-pau.fr/

last update: March 30th, 2015.

Introduction

There are a number of image sensor boards available or proposed by the very active research community on image and visual sensors: Cyclops, MeshEyes, Citric, WiCa, SeedEyes, Panoptes, CMUcam3&FireFly, CMUcam4, CMUcam5/PIXY, iMote2/IMB400, ArduCam, ... All these platforms and/or products are very good and our motivations in building our own image sensor platform for research on image sensor surveillance applications are:

1. have an off-the-shelf solution so that anybody can reproduce the hardware and software
   1. use an Arduino-based solution for maximum flexibility and simplicity in programming and design
   2. use a simple, affordable external camera to get RAW image data, no JPEG
2. apply a fast and efficient compression scheme with the host microcontroller to produce robust and very small image data suitable for large scale surveillance or search&rescue/situation awareness applications
3. simple enough to demonstrate our criticality-based image sensor scheduling propositions
   1. see our paper : C. Pham, A. Makhoul, R. Saadi, "Risk-based Adaptive Scheduling in Randomly Deployed Video Sensor Networks for Critical Surveillance Applications", *Journal of Network and Computer Applications (JNCA), Elsevier, 34(2), 2011, pp. 783-795*
4. easy integration with our test-bed for studying data-intensive transmission with low-resource mote platforms (audio and image)
   1. see our paper: C. Pham, "Communication performances of IEEE 802.15.4 wireless sensor motes for data-intensive applications: a comparison of WaspMote, Arduino MEGA, TelosB, MicaZ and iMote2 for image surveillance", *Journal of Network and Computer Applications (JNCA), Elsevier, Vol. 46, Nov. 2014*
5. fully similar/compatible with our simulation environment based on OMNET++/Castalia for video/image sensor networks.

Architecture and components

We use both Arduino Due and MEGA2560. The Arduino Due board has enough SRAM memory (96KB) to store an 128x128 8-bit/pixel RAW image (16384 bytes). On the MEGA2560, which has only 8KB of SRAM memory, we store the captured image on an SD card (see right figure below for an exemple) and then perform the encoding process by incrementally reading small portions of the image file.

For the camera, we use the uCamII from 4D systems. You can download the reference manual from 4D system web site. The uCamII can deliver 128x128 raw image data. JPEG compression can be realized by the embedded micro-controller but this feature is not used as JPEG compression is not suitable at all for lossy environments. We instead apply a fast and efficient compression scheme with the host microcontroller to produce robust and very small image data.

http://cpham.perso.univ-pau.fr/WSN-MODEL/tool-html/imagesensor.html

Question?

Congduc.Pham@univ-pau.fr