Energy-efficient Operation of GSM-connected Infrared Rodent Sensor

Gábor Paller and Gábor Élő
Széchenyi István University, Information Society Research & Education Group, Egyetem tér 1., Győr, Hungary

Keywords: Agriculture, Infrared Camera, Power Efficiency.

Abstract: Camera sensors have been deployed in the agriculture for various use cases. Most of the applications tried to infer the health and development of the plants based on image data in different wavelength domains. In this paper, we present our research of rodent population estimation with infrared camera sensors.
The usual camera sensor applications in the agricultural domain are quite simple from the sensor architecture point of view as the environment rarely changes. Image capture/transmission at preconfigured moments is usually enough. Rodents move quickly so the sensor must be able to capture images with low capture time interval. As the data link to the server backend is relatively slow, this fast capture rate may require image processing capability in the sensor.
The paper analyzes the effects of such an image processing capability, in particular the power consumption trade-offs. Inadequate power management support of the selected embedded Linux platforms is identified as a problem and proposals are made for improvement.

1 INTRODUCTION

Data captured by agricultural sensors may be scalar (Adamchuk et al., 2004), (López et al., 2009) (Armstrong et al., 1993) or of more complex data types, e.g. spectrogram. Among the latter, 2D imaging sensors ("cameras") have been found to be efficient in detecting effects of drought (Grant et al., 2006), (Alderfasi and Nielsen, 2001), plant phenotype (Li et al., 2014) or diseases (Moshou et al., 2004).

AgroDat.hu is an ongoing research project funded by the Government of Hungary (Paller et al., 2015). In the first phase the project concentrated on sensors providing scalar value like soil temperature, soil moisture, concentration of salts in groundwater and CO_2 concentration in the ground, air temperature, humidity, rainfall, wind speed and direction, solar radiation intensity and leaf wetness. The project decided to use GSM/GPRS network as its data carrier.

Due to the fact that one type of our sensor poles has only underground parts, solar cells as energy source could not be employed hence energy-efficiency was an important concern from the beginning. These decisions made in the first phase when the project deployed only sensors providing scalar time-series data influenced the second phase when cameras are added. In particular, we wanted to reuse the physical sensor pole structure and the GSM/GPRS data carrier layer. This led us into researching energy efficiency in case of the data capture and transmission patterns of 2D image structures.

2 COMMON VOLE DETECTION

Our intention was to create a flexible framework for camera sensors therefore we looked for a more demanding use case. Simple use cases of agricultural camera sensors normally involve taking a picture 1-4 times a day about a specific feature of the environment, e.g. the selected part of the foliage. From the communication pattern and sensor framework points of view, these use cases are just slightly different from the scalar value use case. The sole difference is the larger data size but as these larger data packets are infrequently sent, the conclusions are not significantly different from the ones presented in (Paller et al., 2015).

One of the more challenging use cases we identified is animal monitoring, specifically rodent tracking. Population outbreaks of certain rodent species can cause significant damage in crop production. Grain Producers Association-Hungary (GPAH) estimated that common vole (*Microtus arvalis*, Figure 1) caused 500000 tons of damage in winter wheat alone in 2014. More agressive rodenticides are applied according to population estimation hence this estima-

Paller, G. and Élő, G.
Energy-efficient Operation of GSM-connected Infrared Rodent Sensor.
DOI: 10.5220/0005628500370044
In *Proceedings of the 5th International Confererence on Sensor Networks (SENSORNETS 2016)*, pages 37-44
ISBN: 978-989-758-169-4

Figure 1: Common vole (Microtus arvalis) (source: National Forestry Association, Hungary).

tion is an economically important task.

Common vole population estimation was reported using radio collars (Jacob and Hempel, 2003) but obviously this method is not suitable to detect animals that have not been trapped previously. Detection of itinerant animals which just try to settle on a certain field is best accomplished with cameras but the nocturnal nature of the animal and the excellent camouflage of their fur make them difficult to spot in visible light. (Ou-Yanga et al., 2011) reports about the observation of laboratory animals with infrared camera. The monitored area was rather small - about 50x100 cm - and they employed Kinect sensor. Kinect operates in the near-infrared range (830nm) therefore it requires external illumination of the target which limits its range.

Figure 2 shows a small rodent (*Phodopus sungorus*) similar in size to the common vole in near-infrared image. The picture was taken in complete darkness, the target was illuminated by near-infared light. Note that even though the animal is recognizable, it would be hard to create a computer algorithm that spots it. In this image the animal is close to the camera but as the distance grows, illuminating the target becomes more and more complicated.

Long-wavelength infrared (LWIR) cameras detect the infrared radiation emitted by the object in the picture hence they do not require infrared illumination of the target. LWIR cameras has existed for a long time but their price and other restrictions (e.g. export control) confined them to specialist use cases. Relatively low cost LWIR cameras appeared just recently. We experimented with FLIR Lepton camera module [1] whether small rodents can be detected reliably. This camera operates in the 8000-14000 nm wavelength range and has a resolution of 80x60 pixels. An animal similar to the common vole (*Phodopus sungorus*) was

[1] http://www.flir.com/cores/content/?id=66257

Figure 2: Small rodent similar to a common vole (Phodopus sungorus) in near-infrared image. The animal is marked by the red ellipse.

placed in a cage and images were captured with different distances between the camera and the animal. The background was lawn and other common foliage. The images were made in the night.

Figure 3 shows the result. Minimum and maximum intensity values were calculated from raw image (automatic gain control (AGC) switched off) and the 0-255 pixel intensity in the resulting greyscale image was mapped into this minimum-maximum range so that

$$p_{greyscale} = 255 \frac{p_{raw} - p_{raw,min}}{p_{raw,max} - p_{raw,min}} \quad (1)$$

where p_{raw} is the intensity value of the raw image pixel produced by the Lepton camera, $p_{raw,max}$ and $p_{raw,min}$ are the maximum and minimum intensities detected in the raw image, $p_{greyscale}$ is the pixel intensity value produced for the greyscale image.

When the animal is farther from the camera, its observed infrared radiation decreases as the size of the animal gets closer to the size of a pixel in the image. The advantage of the dynamic calculation of the pixel intensity range is that the animal remains bright, even if it is farther from the camera. Disadvantage is that this method makes features in the background with higher infared radiation (trees, bushes, etc.) to become more pronounced (”brighter”) in case of larger distances to the target which makes the identification of the animal in the image more challenging. The animal is clearly visible up to 4 meters of distance, in fortunate situation the distance can be as large as 5 meters.

The experiments made with the FLIR Lepton LWIR camera convinced us that small animals can be reliably recognized with infrared camera, even with such a limited resolution as the FLIR Lepton has. Due to the non-trivial infared signature of the background, however, we had to develop an image processing algorithm to identify the pictures which are suitable for population estimation.

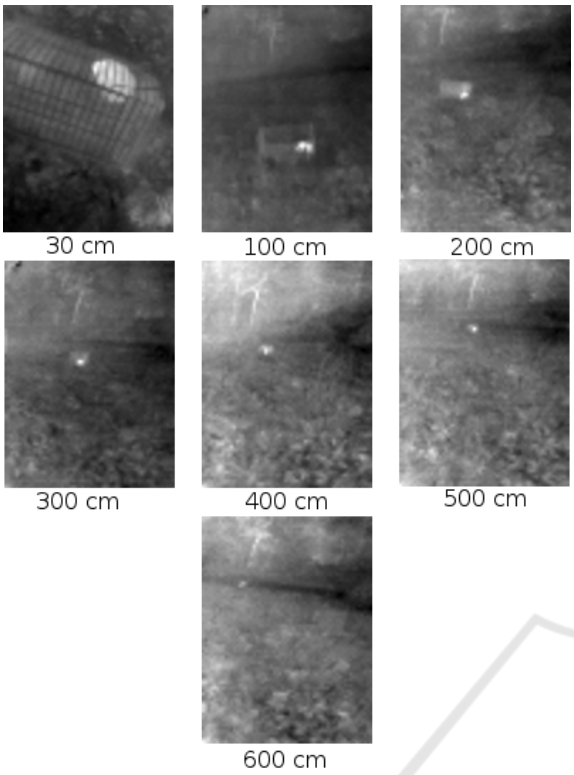

Figure 3: Small rodent similar to a common vole (Phodopus sungorus) in long-wavelength infrared image.

3 IMAGE PROCESSING ALGORITHM IN THE SENSOR

According to the architectural goal of the AgroDat.hu project, data processing happens on the server side. The sensor unit, however, is responsible of sending data that can be meaningfully used to extract relevant information, in our case the population estimate of the small rodents in the area. These animals move relatively quickly so for meaningful observation, the bait area needs to be monitored with an interval of some seconds. Sending an image with this frequency is infeasible for mobile bandwidth and power consumption reasons. The sensor itself must make the first filtering and upload only pictures which has high probability of containing relevant information.

The requirements of the image processing algorithm are the following.

- Remove large non-moving elements from the image. These are supposed to be background elements.
- Look for relatively small objects that are moving. These are potentially the rodents we are looking for.

Our implementation is based on OpenCV image processing framework [2] and comprises the following steps:

- A greyscale image is produced from the raw Lepton output image according to the transformation described in section 2.
- The greyscale image is transformed into a binary image with a fixed threshold of 204 (0.8*255). As the greyscale intensity values are calculated dynamically, taking into account the minimum and maximum intensity values in the raw image, this step is really dynamic thresholding with respect to the original raw infrared image. The high thresholding limit is due to the assumption that our rodents are among the warmest things in the night scene in the agricultural field.
- Contour tracing algorithm from the OpenCV library is applied, then the resulting contours' convex hull is filled. This step gets rid of spurious noise in the image resulting from the thresholding step.
- Elements in the image are dilated by a kernel of 6x6, then again contour traced. This step merges features that are separated by just a gap of up to 6 pixels.
- The encloding circle of each resulting contour is calculated.
- The enclosing circles of this iteration and the previous iteration are compared. In order for two enclosing circles to be considered the same, their overlapping area must be at least 70% of the smaller circle's area. Circles present in both the current and the previous picture are removed from the set of circles in the image. This filtering step ensures that the feature we are looking for needs to move.
- If there are circles remaining that have not been considered the same as any circle in the previous image and the any of the remaining circle's radius is smaller than 5 pixels then we have a candidate image for uploading to the server.

Figure 4 demonstrates the steps of the image processing algorithm. The image labelled as "eq" is the input greyscale image. "th" is the result of the thresholding, note the large amount of unconnected dots. "c1" is the result of the first contour tracing-convex hull filling step. "c2" is the output of the dilation-second contour tracing. At the end, "circle" shows the resulting object circles identified in the picture. Of the 5 circles identified, only one corresponds to

[2] http://opencv.org/

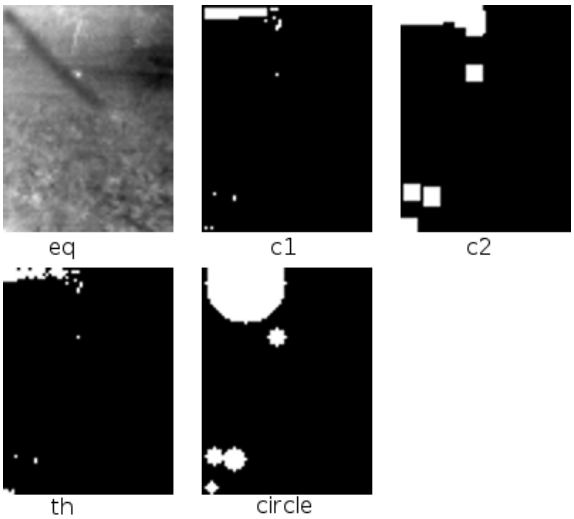

Figure 4: Demonstration of image processing steps in the sensor. Image labels are detailed in section 3.

the test rodent. The rodent is identified when it starts to move. Then its small circle will not overlap with the circle on the previous picture, triggering an image upload.

The algorithm above is presented as a demonstration that the image processing expected from the sensor is not trivial and efficient implementation is heavily based on a publicly available library (OpenCV). As we will see, these conclusions will have important consequences when the power consumption of the sensor is analyzed.

4 THE AGRODAT.HU SENSOR NETWORK INFRASTRUCTURE

As the first version of the AgroDat.hu sensor network will target corn, typical cornfield locations were considered when designing the communication architecture. Due to large field sizes and the production area often located far from existing infrastructure, only a radio technology with large coverage area was acceptable. There are a number of alternative radio technologies with this characteristic (e.g. WiMax or custom VHF/UHF system) but due to its wide availability, low cost and well-established regulatory framework, we decided to use the GSM mobile network.

The first version of the sensor network collects data that change slowly (e.g. soil temperature, soil moisture) and the data representation requires only short data packets (with our coding format it means 200-400 bytes of data). This means that the sensor communicates on the mobile network relatively rarely (1-3 times a day) and even then only low amount

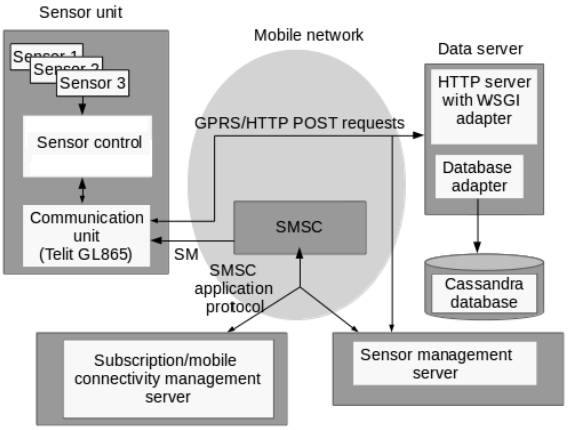

Figure 5: Conceptual communication architecture of the AgroDat.hu sensor network.

of data is sent. Part of the sensor stations is only equipped with underground sensors and minimally protrude above ground level therefore solar cell-based power supply was not possible. Energy efficiency was a key requirement when designing the sensor station. Due to the low amount of data transfer in the first version and the energy efficiency requirement, the prorotype was designed using the low-bandwidth services of the GSM network. This may mean GPRS or SMS-based data transfer.

The second iteration added image and video transfer to the requirement set. This new requirement changes the amount of data to transmit. One infrared image is typically in 3-4 kByte of size (80x60 pixel resolution, 16 bit greyscale, PNG format). The sensor may also send short videos (20-30 sec) created from consecutive infared images. In MP4 format, these videos have the typical size of 30-50 kBytes. Images captured by ordinary web cameras are up to 60 kBytes in PNG or JPEG format. This higher payload size may require the usage of more recent telecommunication technology, e.g. 3G or 4G. Our experience, however, is that in the deployment areas of our sensors, even basic GSM coverage may be problematic, particularly with an antenna mounted so close to the ground that some of our sensors have. For the experiments reported in this paper, we did not change the GSM/GPRS communication technology but analyzing the possibility of 3G/4G is definitely a work to be done.

Figure 5 shows the architecture used in the second iteration of the AgroDat.hu sensor network. The GPRS/HTTP is used to send bulk data to the server. (Paller et al., 2015) demonstrated that the use of protocols more optimized than HTTP like CoAP does not yield a significant power consumption saving over the GPRS bearer. The SMS infrastructure is employed as the push bearer for server-initiated operations like

management operations. (Paller et al., 2015) analyzed the energy-efficiency of two ”push” solutions and found that SMS is significantly more energy-efficient than long-lived TCP connections. The management server initiates management operation by sending SMs to the sensor. Some of these commands like status query, changing a single configuration variable, reset request fit into one SM and the entire operation is carried out over the SM service. In case of other commands like sensor log download or software update, the SM sent by the management server is used to trigger a GPRS connection to the management server and the data transfer is carried out over the GPRS connection. The main change compared to the soil sensor is the sensor control component. In the first iteration, this component controlled a set of low-bandwidth sensors with scalar value output therefore an Atmel ATxmega128A4 microcontroller performed the sensor control task. As argued in section 3, for the camera sensor case we have a set of requirements that demands more powerful sensor control. These requirements are the following.

- In case of the cameras operating in visible light frequency domain, we want a platform that facilitates the interfacing of popular cameras, e.g. USB-connected webcams.
- We need a platform that is powerful enough to execute the relatively complex image processing tasks detailed in section 3.
- In order to implement the image processing logic efficiently, we need a platform that is able to deploy popular image processing frameworks like OpenCV.

Considering these requirements led us to the conclusion that the sensor control component in our camera sensor should be implemented based on an embedded ARM-based computer (we used BeagleBone Black, based on the TI Sitara AM335x ARM Cortex-A8 system-on-chip (SoC)) and an embedded Linux operating system. BeagleBone Black is supported by a number of Linux distributions. The measurements were made with 2 of them: Ubuntu Snappy [3] which is a variant of the Ubuntu distribution particularly targeted to Internet of Things (IoT) applications and Texas Instrument's own version of Linux specifically targeted to the Sitara family of SoCs, called Linux EZ Software Development Kit [4]. Snappy is attractive due to the large software base that Canonical, the developer of the Ubuntu distribution constantly updates. For example OpenCV and the libraries it depends on are available as software packages for this platform. TI EZSDK on the other hand provides the best available integration with the Sitara SoC's hardware features of which the power management was particularly important for us.

[3]https://developer.ubuntu.com/en/snappy/

[4]http://www.ti.com/tool/linuxezsdk-sitara

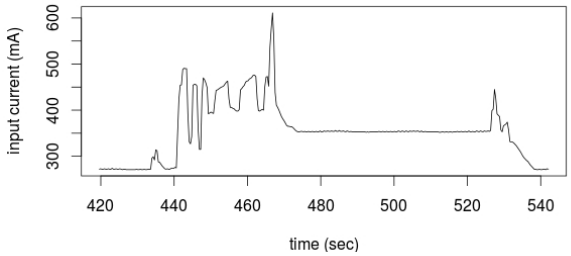

Figure 6: Image processing-image sending cycle in case of communication logic running on the Sitara CPU.

5 IMAGE PROCESSING AND POWER CONSUMPTION

It has been already pointed out in the literature (Margi et al., 2006) that the power consumption of a camera sensor node can be chategorized as idle, processing-intensive, storage-intensive, communication-intensive and visual sensing. In this section we try to optimize the power consumption by allocating idle consumption to processing units and by finding a balance between processing-intensive and communication-intesive tasks.

In the first version of the AgroDat.hu sensors the main control logic of the sensor was executed by the Telit GL865 module which combines the GSM modem with a low-power microcontroller. As Snappy Ubuntu is positioned as a platform with secure update option, the first sensor architecture we evaluated was that both the image processing and the communication logic was executed by the main Sitara CPU running Snappy Ubuntu Linux, the Telit GL865 was used only as a modem. Figure 6 shows the power consumption of the entire system (BeagleBone Black and the GL865 GSM modem) when running the image processing-image sending cycle. The image processing cycle happens between the timestamps of 435-440 seconds while the image sending is between the timestamps of 440-540 seconds. The image size to be sent to the server with a HTTP POST request was 4 Kbytes, over GPRS bearer.

The resulting power consumption for the image sending cycle is 10.755 mAh which is much higher than the expected consumption of about 1 mAh in (Paller et al., 2015). The source of this significant difference is the Sitara CPU's base non-idle consumption of 250 mA. This alone results in 250 mA*100

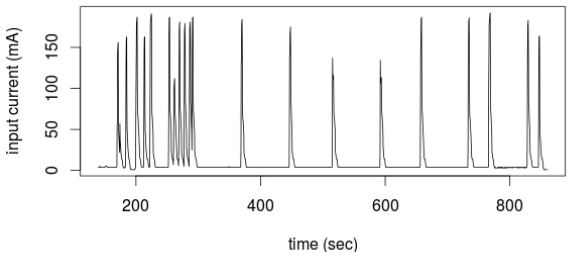

Figure 7: Image sending cycle in case of communication logic running on the GL865 module.

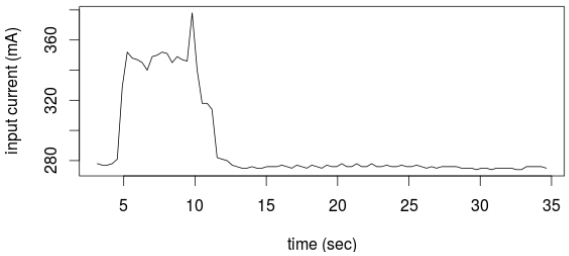

Figure 8: Image processing power consumption, 5 iterations, 500 msec inter-image delay.

seconds=25000 mAs (about 7 mAh) consumption. This example shows that the high-performance Sitara performs very poorly from the power consumption point of view if the computation pattern is of action-and-wait type which is very common in case of telecommunication state machines. In this case the event the application code waits for happens too quickly therefore the CPU cannot be put into an idle state while the overall length of the execution is quite long. This level of power consumption overhead was unacceptable for us, hence we decided to rearchitect the sensor so that the code is executed on a CPU that is most suitable for the task with regards to power consumption.

In the rearchitected model we returned to the original sensor control model where the main control logic is in the GL865 module. The microcontroller in the GL865 is very efficient in low-power, low-performance execution and can sleep with very low power consumption, using its internal real-time clock. The experience also applies to different communication modules where the controller and the modem are not integrated. In this case a separate low-power microcontroller can take care of the main control. Due to its high active power consumption, the Sitara CPU is kept in inactive state as long as possible.

Figure 7 shows the power consumption of the GL865 sending the image of 4 Kbytes. The sending cycle takes much longer (600 seconds vs. the 100 seconds when the Sitara CPU executed the same logic), this is due to the slowness of the GL865 Python engine. The power consumption for the sending cycle is much lower, however, about 3 mAh while the power needed to send over the image from the Sitara CPU to the GL865 over the serial line is negligible.

In case of the rearchitected sensor control, the Sitara CPU waits for wakeup signal, acquires and processes images and if suitable image is available, uploads the images to the GL865 over the serial connection between the two units. Then the Sitara CPU goes to sleep. Currently we see the following use cases for the combinations of timed picture acquisition vs. automatic identification of relevant pictures.

- Images or video (infrared and/or visible-light) are acquired at preconfigured moments of time. In this case the advantage of the embedded Linux in the sensor is its wide support of different cameras. Image processing is not done.
- At preconfigured moments of time, automatic image acquisition is triggered and if image processing finds a relevant feature, the images/videos (visible light/infrared) are uploaded to the server.
- Image acquisition/image processing is executed continuously and images are uploaded to the server if relevant feature is found by the image processing pipeline. There are sub-requirements about the acquisition time interval between the images (from zero to up to 60 seconds) but the images are not sent to the server at preconfigured moments, only when the image processing algorithm identifies something important.

Depending on the use case, there are different power consumption balance between the image processing and image sending activities. For the first use case that sends images only at preconfigured moments, the use of high-performance SoC platform with embedded Linux is not justified from the power consumption point of view. There is some marginal software engineering advantage as embedded Linux comes with a large number of camera libraries and image acquisition tools (e.g. V4L). Regarding the third use case, image processing runs continuously therefore this function consumes the most power, there is no point in talking about power consumption balance. Power consumption balance becomes relevant in the second use case where images taken may not be sent if there is no relevant feature on them. The image processing performed to figure out if the image is worth sending may be justified by lower amount of data to transfer and lower power consumption as GSM communication is an expensive operation from the power consumption point of view. Prior literature about camera sensor power consumption has already pointed out that tasks with lower energy consumption should trigger tasks with higher energy consumption

(Kulkarni et al., 2005) and in this case the image processing corresponds to the task in with lower hierarchy level to trigger the GPRS transfer.

Figure 8 shows the power consumption of an image acquisition/processing activity on Ubuntu Snappy 15.04. One iteration comprises image acquisition from the infrared camera, running the algorithm described in section 3 and waiting 500 msec if no relevant feature was found. The image acquisition/processing was repeated 5 times, consuming about 0.62 mAh. It can be observed that the power consumption increased for the duration of the activity as clock scaling option was active (cpufreq with "ondemand" governor). This automatically increased the clock speed to the maximum of 720 MHz from the base of 275 MHz when the CPU was active. Even though the Sitara SoC contains a GPU, this time it was not used because the small size (80x60) of the infrared images would not make the GPU usage efficient.

Considering strictly the image acquisition/image processing step, the 0.62 mAh consumption of this 5-iteration activity compares favorably to the 3 mAh consumption of sending the image with the GL865. The Sitara CPU has a significant idle consumption, however. Our prototype was implemented on the Ubuntu Snappy distribution which at the moment of writing this paper, does not offer CPU idling support. This means that an inactive CPU still consumes about 250 mA (same as active CPU with no load), consuming 3 mAh (the cost of sending one image) in just 43 seconds. The TI EZSDK implements one sleep state, the suspend-to-RAM (S3) state. TI EZSDK can enter and exit this state in 3 seconds but the consumption in this state is still 156 mA, which means 69 seconds to reach 3 mAh. Ubuntu Snappy 15.04 consumes 120 mA even in shutdown state but TI EZSDK properly shuts down. Unfortunately, a full shutdown-reboot consumes 4.78 mAh with TI EZSDK which is more than the 3 mAh required to send the image. Idling the Sitara CPU with shutdown is therefore not an option.

Our conclusion is that saving battery power and cellular data transfer by putting more intelligence into the sensor and prefiltering image data there is still an attractive option. Unfortunately the current platforms are inadequate from the power consumption point of view, in particular the idle state management needs more improvement. Until the embedded Linux sytem can be placed into a state with near-zero consumption in relatively short time, efficient battery-powered operation is not possible.

We found that a use case exists for applications with image processing in the sensor. If the requirement is to monitor the environment continuously with short image capture interval and the data link to the server side is relatively slow, detection of the relevant features must be done in the sensor. This was the case for our rodent detection use case. According to our experiments, if the image processing is implemented on the BeagleBone Black using high-productivity, popular software stacks (e.g. Linux/OpenCV), the power consumption will be very high. It is certainly possible to decrease this high power consumption with dedicated hardware (e.g. microcontrollers) but the software productivity will drop dramatically as powerful image processing frameworks are not available for these devices. The outcome is that continuous monitoring with high-productivity frameworks is an expensive choice from the power consumption point of view.

6 CONCLUSIONS

Camera sensors have been deployed in the agriculture for various use cases. Most of the applications tried to infer the health and development of the plants based on image data in various wavelength domains. These applications are simple from the sensor point of view as capturing/sending images at predetermined moments is usually enough. The larger data payload that these sensors generate would justify the usage of a more recent cellular standard (3G/4G) but coverage is spotty in the areas of our interest. We intend to analyze more the question of 3G coverage in areas relevant for agricultural activity.

In our research, we looked for a use case that requires more sophisticated processing in the sensor and we found that rodent population estimation is an economically relevant application and due to the quick movement of the target animals, fast capture interval is required. We also found that a bait area can be efficiently monitored with a reasonably priced long-wavelength infrared camera.

It was an attractive proposition that the power consumption of the sensor system can be efficiently decreased with image processing because the sensor can filter out non-relevant images. Initial analysis of power consumption cost of a relatively complex image processing operation was promising. Unfortunately the idle state support of the embedded Linux platform of our choice prevented the exploitation of this possibility. We found that continuous image capture/monitoring is a use case that still requires image processing capability in the sensor but energy-efficient implementation is not supported with the popular software stack we evaluated (Linux/OpenCV). There is a trade-off here between implementation productivity and energy effi-

ciency with these platforms.

We propose the following directions to resolve this trade-off. One direction is to improve the idle state management of embedded Linux systems. This is not trivial due to the complex interaction between the peripherals (e.g. network cards) and the tasks running on the main processor. The expectation here is that the Linux system may be placed into a state where it consumes near to zero current. The other direction is to port software libraries necessary for image processing to microcontrollers without operating systems. This would eliminate Linux and its power management complexity entirely from the picture but it would also make impossible the exploitation of security management and prepackaged software of embedded Linux systems.

ACKNOWLEDGEMENTS

I would like to thank the Government of France for the generosity in granting a scholarship to the ESEO institute in Angers, France. I thank especially Sébastien Aubin at ESEO for facilitating my stay.

REFERENCES

Adamchuk, V. I., Hummel, J., Morgan, M., and Upadhyaya, S. (2004). On-the-go soil sensors for precision agriculture. *Computers and electronics in agriculture*, 44(1):71–91.

Alderfasi, A. A. and Nielsen, D. C. (2001). Use of crop water stress index for monitoring water status and scheduling irrigation in wheat. *Agricultural Water Management*, 47:69–75.

Armstrong, R., Barthakur, N., and Norris, E. (1993). A comparative study of three leaf wetness sensors. *International journal of biometeorology*, 37(1):7–10.

Grant, O. M., Chaves, M. M., and Jones, H. G. (2006). Optimizing thermal imaging as a technique for detecting stomatal closure induced by drought stress under greenhouse conditions. *Physiologia Plantarum*, 127(3):507–518.

Jacob, J. and Hempel, N. (2003). Effects of farming practices on spatial behaviour of common voles. *Journal of Ethology*, 21(1):45–50.

Kulkarni, P., Ganesan, D., Shenoy, P., and Lu, Q. (2005). Senseye: a multi-tier camera sensor network. In *Proceedings of the 13th annual ACM international conference on Multimedia*, pages 229–238. ACM.

Li, L., Zhang, Q., and Huang, D. (2014). A review of imaging techniques for plant phenotyping. *Sensors*, 14(11):20078.

López, J. A., Soto, F., Sánchez, P., Iborra, A., Suardiaz, J., and Vera, J. A. (2009). Development of a sensor node for precision horticulture. *Sensors*, 9(5):3240–3255.

Margi, C. B., Petkov, V., Obraczka, K., and Manduchi, R. (2006). Characterizing energy consumption in a visual sensor network testbed. In *Testbeds and Research Infrastructures for the Development of Networks and Communities, 2006. TRIDENTCOM 2006. 2nd International Conference on*, pages 8–pp. IEEE.

Moshou, D., Bravo, C., West, J., Wahlen, S., McCartney, A., and Ramon, H. (2004). Automatic detection of yellow rustin wheat using reflectance measurements and neural networks. *Computers and electronics in agriculture*, 44(3):173–188.

Ou-Yanga, T.-H., Tsaib, M.-L., Yenc, C.-T., and Lina, T.-T. (2011). An infrared range camera-based approach for three-dimensional locomotion tracking and pose reconstruction in a rodent. *Journal of Neuroscience Methods*, 201:116–123.

Paller, G., Szármes, P., and Élő, G. (2015). Efficient power consumption strategies for stationary sensors connected to GSM network. In *SENSORNETS 2015: Proceedings of the 4th International Conference on Sensor Networks., At Angers, France, Volume: pp. 63-68., editors: César Benavente-Peces, Patrick Plainchault, Octavian Postolache*.