CUDA Accelerated X-ray Image Processing for Bone Detection in the Valka X-ray Guided Cutting Machine

Emil Atli Ellegaard,
Sveinn Elmar Magnússon

Computer Science B.Sc

Instructor: Stefán Freyr Stefánsson
Examiner: Torfi Helgi Leifsson

Spring 2015

Abstract

The Valka X-ray Guided Cutting Machine is a fish fillet cutting machine which takes an X-Ray image of every fillet fed to the machine via conveyor. The x-Ray is then joined with a 3D image which gives a three dimensional image of the fillet and its bones. The cutting is made with high pressure water jets. The water jets are controlled by robots which cut out the bones and portion the fillet with high precision while the fillet is rolling on the conveyor. The result is precision cut fillets for maximum revenues.

This project involves rewriting the existing bone detection for the Valka X-Ray Guided Cutting Machine using CUDA to further accelerate the bone detection allowing more time for other calculations such as cutting analysis. The new CUDA based bone detection is then compared to its CPU based predecessor for benchmarking. It also involves researching if OpenCV is a viable option for future development of image processing and recognition for Valka machines.

Contents

1 Introduction

This report is part of a B.Sc. final project in Computer Science at Reykjavík University. The product of this project is a CUDA version of the bone detection software, the Valka X-ray Guided Cutting Machine uses for detecting bones in x-ray images of fish fillets. The project is done in collaboration with Valka ehf, which is a technology company specializing in development and marketing of equipment and automatic solutions for the fish industry.

Firstly Valka wanted to examine if CUDA is a viable option to use in their x-ray guided cutting machine. Since Valka had a problem with the current CPU bone detection library being bit too slow, they approached the team to build a CUDA version of the bone detection software and analyze if CUDA is worth using in their machines. Secondly Valka wanted to explore if OpenCV is a option for further developing their image processing software.

The goal of the project are first and foremost to implement a CUDA version of the current CPU bone detection and compare it to the current CPU version by timing and profiling both versions. In addidtion, the team wanted to examine a OpenCV version of the filters, compare it to Valka's current filters and the new CUDA version of the filters.

The Valka X-ray Guided Gutting Machines is one of a kind on the market. Therefore an x-ray image fish bone detection is not a common problem that has been solved many times before and probably never using CUDA.

Due to confidentiality to Valka, all methods have been renamed methodA to methodZ.

2 Overview

The chapter is split into three sub chapters, where the first two contains background information about Valka ehf and the Valka X-ray Guided Cutting Machine. The third sub chapter contains general information about CUDA computing platform.

2.1 Valka ehf

Valka specializes in the development and marketing of equipment and automation solutions for the fish industry.

Founded in 2003 and based in Kópavogur Iceland, Valka has gained a proven reputation for producing high quality products. The company emphasizes on designing high technology hardware and software aimed at enhancing the customer's productivity through innovative solutions and with effective teamwork.

Representing highly experienced individuals from the technical side of the seafood industry, the Valka team shares the vision and principles of solving the customers mission critical problems promptly and efficiently.

Valka is a world leader in fish cutting technology and was the first in the world to market an efficient pin-bone cutting machine and five machines are now in use in Norway and Iceland. Valka currently holds five patents with more pending and has 32 employees.[1]

2.2 The Valka X-ray Guided Cutting Machine

The Valka X-ray Guided Cutting Machine, seen in figure 1, automatically removes pin bones from fish fillets and cuts them down to desired portions. The machine uses combination of an X-ray and 3D image processing together with robot controlled water jets to locate and cut out pin bones and portions the fillet with exceptionally high accuracy. The Cutting Machine then relays information through the Valka Rapidfish Software to its clients, which are fish processing plants all over the world.[1]

2.3 CUDA

Compute Unified Device Architecture (CUDA) is a parallel computing platform and programming model invented by NVIDIA. It utilizes the power of NVIDIA graphics processing unit (GPU) to dramatically increase computing performance with parallel computing[2].

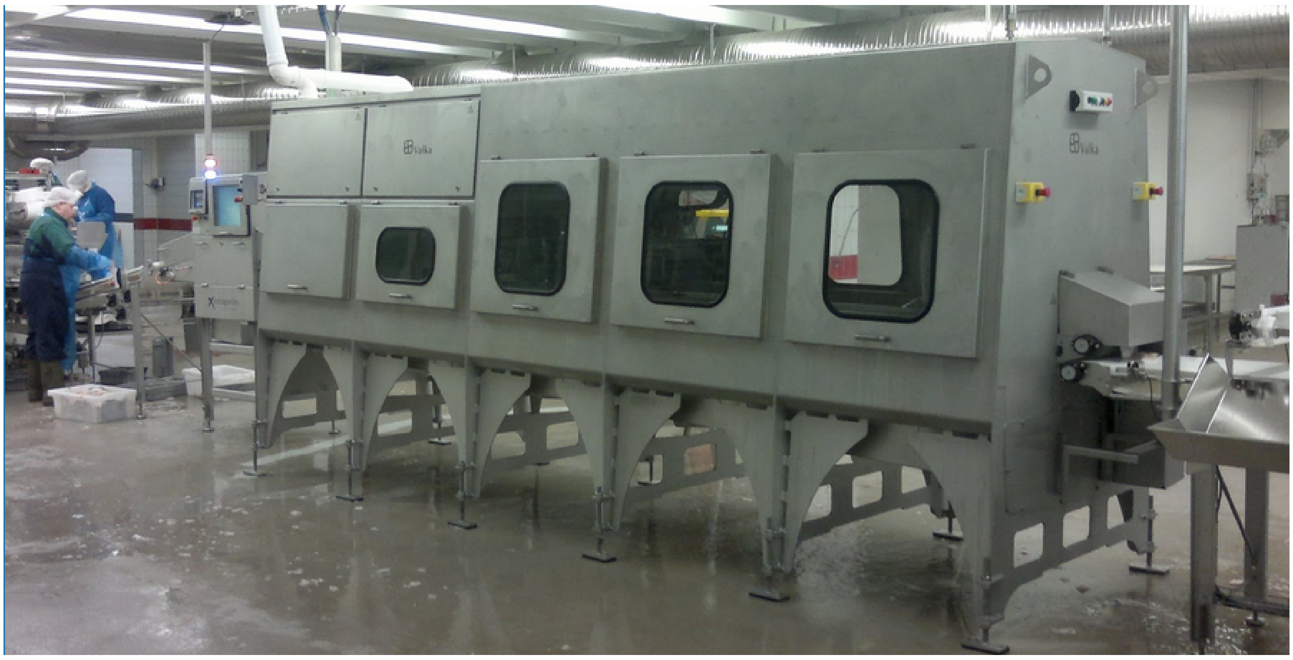


Figure 1: The Valka X-ray Guided Cutting Machine

CUDA enables general purpose processing i.e. not only graphics. This is called General Purpose Graphics Processing Unit (GPGPU). Unlike CPU's GPU's have a parallel throughput architecture, which can execute multiple threads relatively quickly while CPU's can execute a single thread very quickly[2]. When dealing with large arrays such as images, the GPU can give considerable speed increase by executing calculations for multiple pixels simultaneously.

2.4 OpenCV

Open Source Computer Vision (OpenCV) is a library of programming functions mainly aimed at real-time computer vision and is free for use under the open source Berkeley Software Distribution (BSD) license. The library is cross-platform and focuses mainly on real-time image processing.[3] OpenCV can be built with CUDA support which enables the user to use built in filters.

3 Objective

The Valka X-ray Guided Cutting Machine, seen in chapter and 2.2 figure 1, uses X-ray detector to locate bones in fish fillets. The bones are then cut out of the fillets with water jets pressurized to 3.000 bar. The cutting is done with robots and the fillets move on a conveyor while the cutting takes place. Along with bone detection the machine is able to portion each fillet for maximum revenues. This requires fast bone detection and calculations for increased efficiency and accuracy. Figure 2 shows X-ray image of a cod fillet.

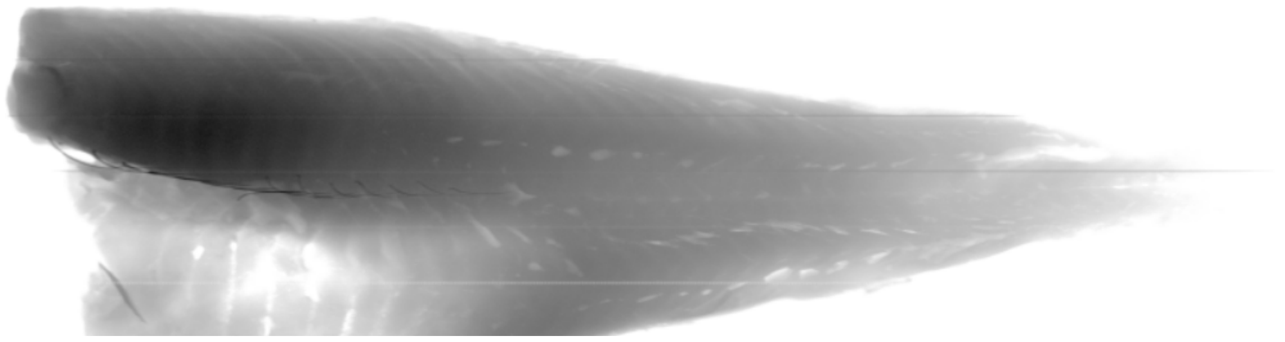


Figure 2: X-ray image of a cod fillet

Current software is written in C++ and C#. Current bone detection software has been implemented in C++ using OpenMP for acceleration running on eight CPU threads. Further information on OpenMP can be found at `http://openmp.org/` Current code does not utilize OpenCV and all methods are written from scratch by Valka.

The goal of the project is twofold. Firstly the use of CUDA to accelerate existing bone detection and secondly to see if OpenCV image processing library can be beneficial in further development of the Valka X-Ray Guided Cutting Machine. The use of CUDA to accelerate existing bone detection had higher priority since Valka has a working bone detection not written in OpenCV.

3.1 Acceleration of Valka X-Ray Guided Cutting Machine

As mentioned before, the higher priority objective aims at increasing the speed of current bone detection software by utilizing CUDA. The benefits of increased speed are increased throughput and cutting precision of the machine, less rejection and the possibility of making the machine shorter. The next four sub chapters describe the benefits in detail.

3.1.1 Shorter machine

The benefits of making the machine shorter is increased space on the floor of the fish processing plant along with being able to locate the machine in smaller confinements. This gives processing plants with less floorspace the ability to add the Valka X-Ray Guided Cutting Machine to its arsenal.

3.1.2 Increased throughput

By increasing the throughput leads to increased revenues for the processing plant. For example a typical cod processing plant has a throughput of around 6.000 metric tons a year.

If increased speed of the software leads to 1% increase in throughput of the processing plant and the price of one kilogram of fully processed cod is €8, it would lead to increased revenues of €480.000 for the processing plant for those 60 tons increase.

3.1.3 Increased precision

Increased cutting precision means less fish goes to waste and therefore increased revenues for the processing plant. Future versions of the machine will have 3D cutting for maximum precision.

3.1.4 Less rejection

With faster image processing, less chance is of fillets being rejected because its X-ray image could not be processed in time before the fillet is to be cut. This leads to increased reliability and therefore more overall throughput.

3.2 Examining OpenCV

The lower priority objective is examining if OpenCV is a viable option for the bone detection software. Current software is all written from scratch by Valka. This makes it interesting to benchmark current solution to the new CUDA version as well an OpenCV solution.

3.3 Final Product

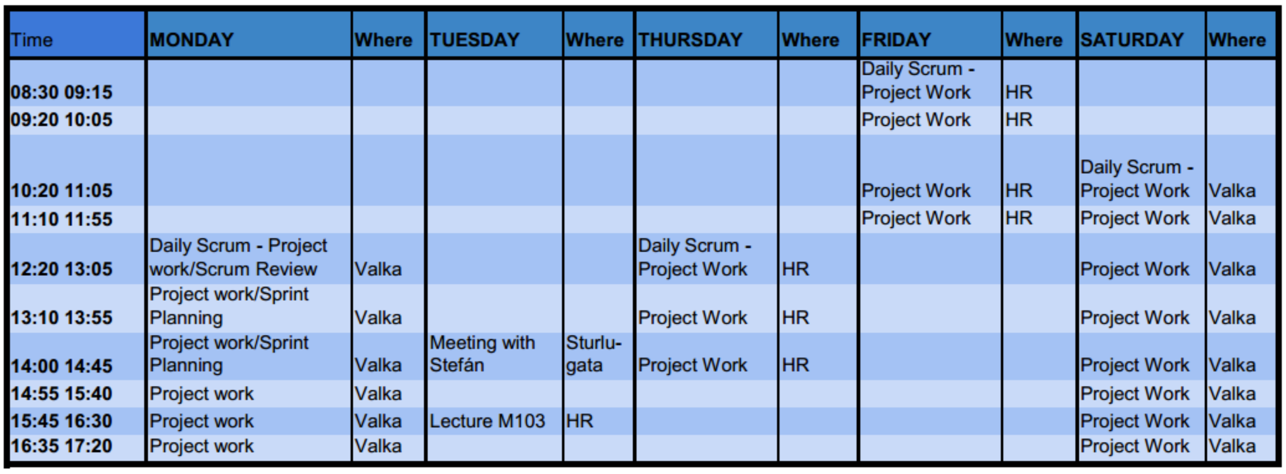
The final product consist of CUDA accelerated version of current bone detection software and report with benchmarking between the original and the CUDA accelerated version. OpenCV version of part of the bone detection compared the original bone detection software. All work, code and reports is a property of Valka. and shall be returned to Valka upon completion.

4 Project Management

The team used Scrum methodology where Sveinn Elmar Magnússon had the role of Scrum Master and Emil Atli Ellegaard the role of product owner. Both team members also worked as programmers. Pair programming was favored since the team had good experience of pair programming from earlier work. The team used Git for version control and Google Spreadsheet used for all project management.

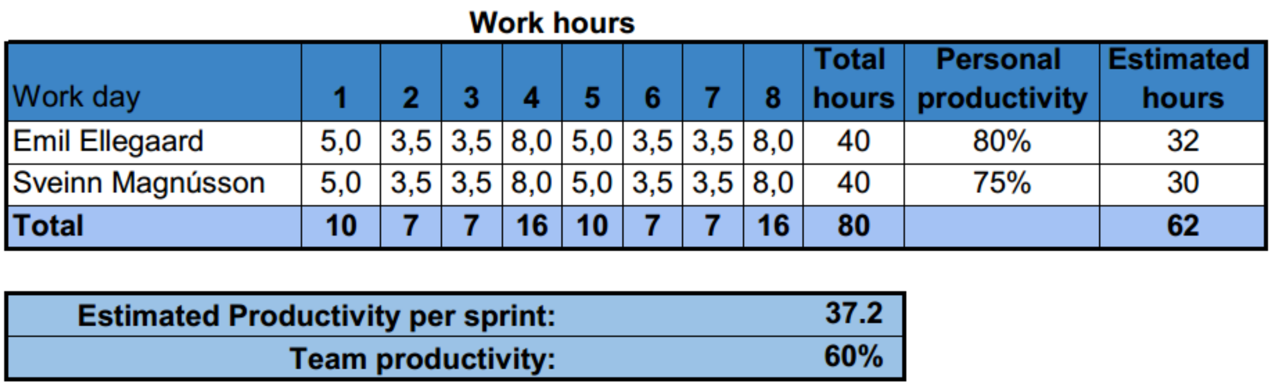
All programming is done in C++ and CUDA C using Microsoft Visual Studio for an Integrated Development Environment (IDE). As stated in chapter 3 OpenCV and CUDA

Toolkit 6.5 was used as third party software. N-sight from NVIDIA was used for profiling along with other tools such as BurnItIn and Isee image viewer.



Time	MONDAY	Where	TUESDAY	Where	THURSDAY	Where	FRIDAY	Where	SATURDAY	Where
08:30 09:15							Daily Scrum - Project Work	HR		
09:20 10:05							Project Work	HR		
10:20 11:05							Project Work	HR	Daily Scrum - Project Work	Valka
11:10 11:55							Project Work	HR	Project Work	Valka
12:20 13:05	Daily Scrum - Project work/Scrum Review	Valka			Daily Scrum - Project Work	HR			Project Work	Valka
13:10 13:55	Project work/Sprint Planning	Valka			Project Work	HR			Project Work	Valka
14:00 14:45	Project work/Sprint Planning	Valka	Meeting with Stefán	Sturlu-gata	Project Work	HR			Project Work	Valka
14:55 15:40	Project work	Valka							Project Work	Valka
15:45 16:30	Project work	Valka	Lecture M103	HR					Project Work	Valka
16:35 17:20	Project work	Valka							Project Work	Valka

Table 1: Weekly time table first 12 weeks



Work hours

Work day	1	2	3	4	5	6	7	8	Total hours	Personal productivity	Estimated hours
Emil Ellegaard	5,0	3,5	3,5	8,0	5,0	3,5	3,5	8,0	40	80%	32
Sveinn Magnússon	5,0	3,5	3,5	8,0	5,0	3,5	3,5	8,0	40	75%	30
Total	**10**	**7**	**7**	**16**	**10**	**7**	**7**	**16**	**80**		**62**

Estimated Productivity per sprint:	**37.2**
Team productivity:	**60%**

Table 2: Team Productivity first 12 weeks

For the first 12 week, each sprint consisted of two weeks with four working sessions per week, on Mondays, Thursdays, Fridays and Saturdays. See table 1 for the weekly timetable. In the 3 week period, the length of each sprint is one week with work hours between 8:30 and 17:30. Total working hours in one sprint for each member was 40 hours both in the first 12 week period and the later three week period.

As mentioned before, for the first 12 weeks, each week was divided into four working sessions as shown in table 1, and total working hours per sprint was 40 hours per team member. As the team was unfamiliar with CUDA, the estimated effiency of the team was 60% at the beginning. Furthermore the programming time for the Scrum master was estimated to be 75% and 80% for the product owner. Therefore the estimated total hours for programming per sprint was 37,2 hours, see table 2. This estimation held in the beginning but as the team grew to know the project better, the productivity increased.

Gantt chart was created to organize time and mark return deadlines of the product, this time plan took into consideration peaks in other courses as well as holidays and exams. Copy of the time plan can be seen in table 3.

Most of the work was done at Valka offices and Reykjavik University. Twice a week the team worked at Valka offices and twice a week at the university. Meetings with Valka and the instructor, Stefán Freyr Stefánsson where held on requests.

5 Risk Management

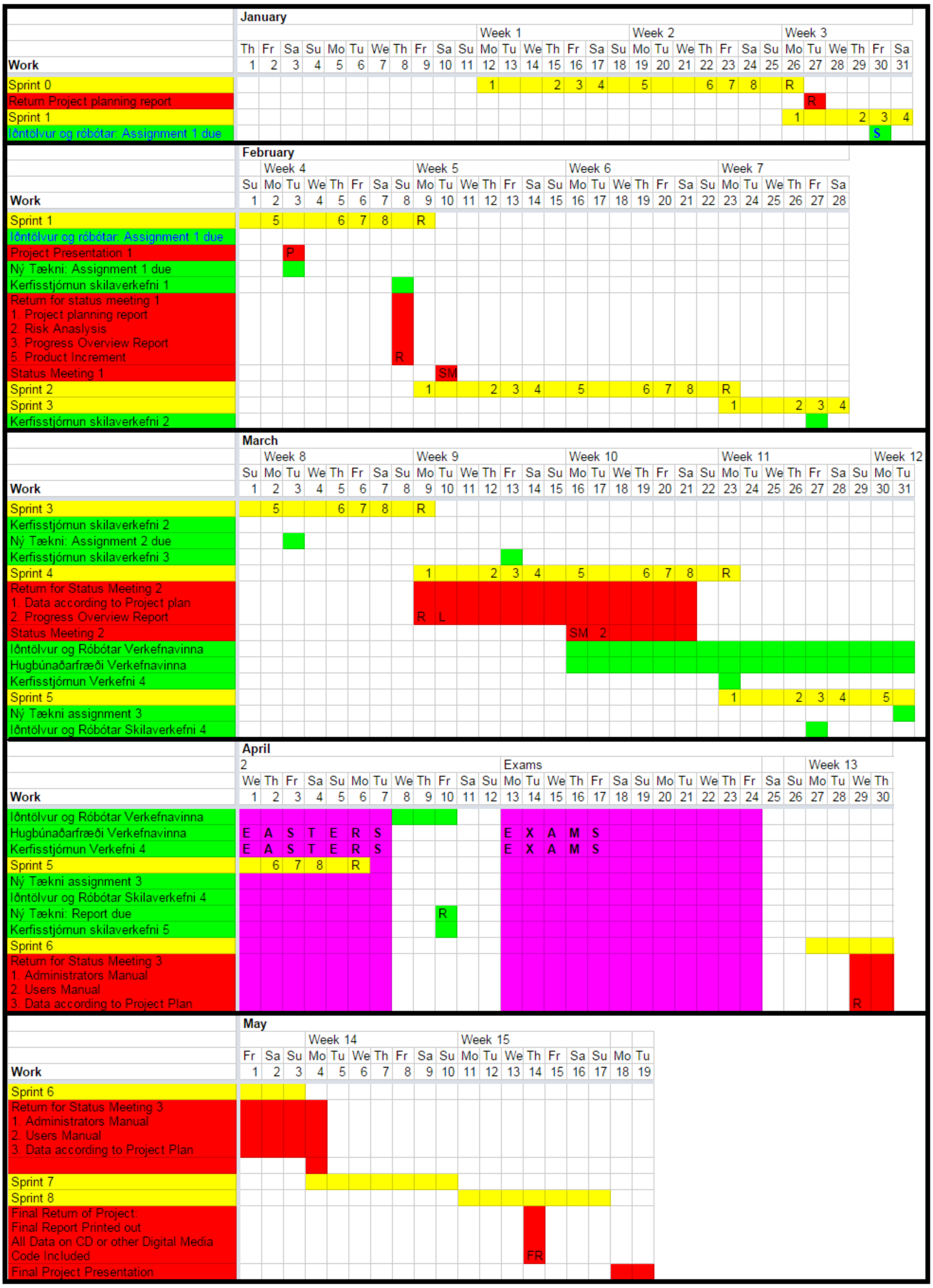
The team encountered few risk factors and was able to minimize the damage of the risk factors with quick and precise reactions. The risk factors the team encountered are listed here below.

- Some changes where made to the product backlog although the days lost where not nearly as severe as predicted. See chapter 6.4 for more information.
- Complexity of the Valka code somewhat slowed down the team caused by the overhead of irrelevant code to the project. This was solved with a new simpler test environment for the new CUDA based bone detection library.
- Reconstruction of the code was necessary because of the slow warmup time of the CUDA library (300 milliseconds for the first cudaMalloc()). This was solved by redesigning the library such that CUDA warmup is done higher up in the hierarchy. See chapter 6 for more information.
- Team members have been ill and other courses took more time than anticipated. This has been solved by working extra hours on other days.

Risk Analysis can be seen in table 4.

6 Progress

This chapter emphasizes on the progress of the project. It discusses product backlog and each sprint in more detail. What goals where set for each sprint and if the goals where met. It also contains information on how the sprints went in general and how much time was spent in each sprint. The current status of the project is discussed and it ends with a work log summary which explains how the team spent their working time.



	January																													
												Week 1							Week 2							Week 3					
	Th	Fr	Sa	Su	Mo	Tu	We	Th	Fr	Sa	Su	Mo	Tu	We	Th	Fr	Sa	Su	Mo	Tu	We	Th	Fr	Sa	Su	Mo	Tu	We	Th	Fr	Sa
Work	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Sprint 0												1	🟨	🟨	2	3	4	🟨	5	🟨	🟨	6	7	8	🟨	R					
Return Project planning report																											R				
Sprint 1																										1	🟨	🟨	2	3	4
Iðntölvur og róbótar: Assignment 1 due																														S	

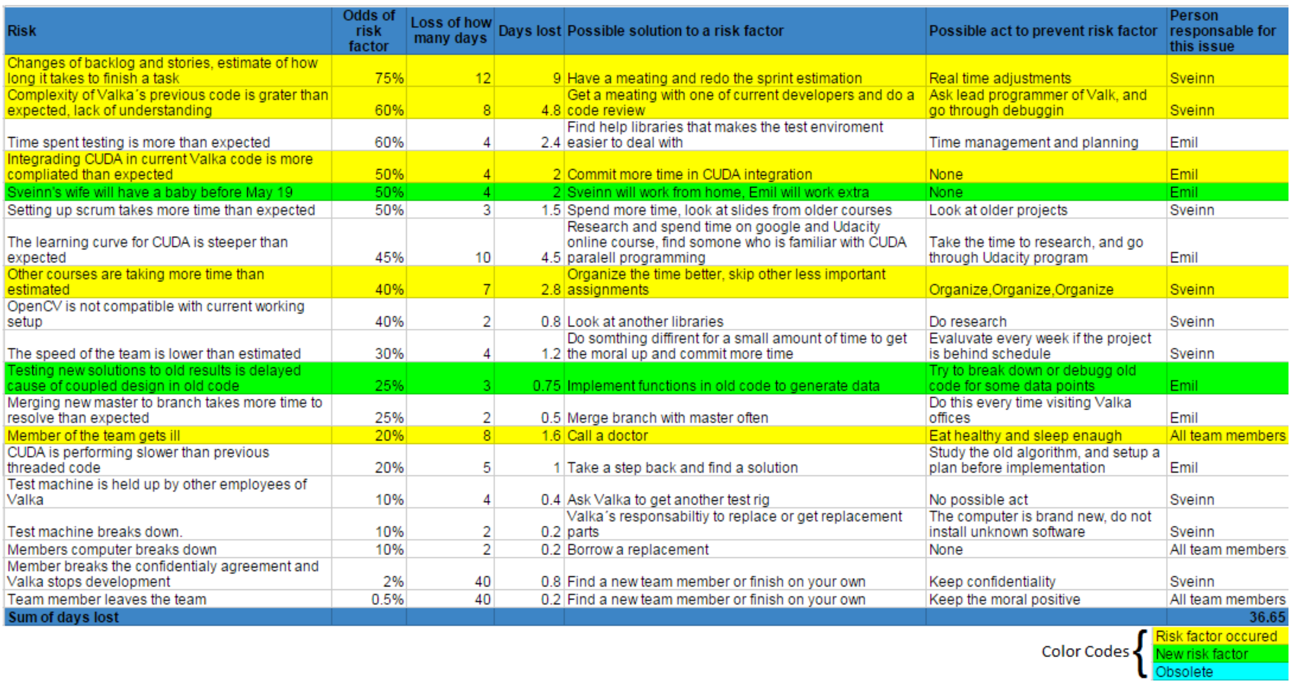
	February																											
		Week 4							Week 5							Week 6							Week 7					
	Su	Mo	Tu	We	Th	Fr	Sa	Su	Mo	Tu	We	Th	Fr	Sa	Su	Mo	Tu	We	Th	Fr	Sa	Su	Mo	Tu	We	Th	Fr	Sa
Work	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28
Sprint 1	🟨	5	🟨	🟨	🟨	6	7	8	R																			
Iðntölvur og róbótar: Assignment 1 due																												
Project Presentation 1			P																									
Ný Tækni: Assignment 1 due			🟩																									
Kerfisstjórnun skilaverkefni 1								🟩																				
Return for status meeting 1 1. Project planning report 2. Risk Anaslysis 3. Progress Overview Report 5. Product Increment								R																				
Status Meeting 1										SM																		
Sprint 2									1	🟨	🟨	2	3	4	🟨	5	🟨	🟨	6	7	8	🟨	R					
Sprint 3																							1	🟨	🟨	2	3	4
Kerfisstjórnun skilaverkefni 2																											🟩	

	March																														
		Week 8							Week 9							Week 10							Week 11							Week 12	
	Su	Mo	Tu	We	Th	Fr	Sa	Su	Mo	Tu	We	Th	Fr	Sa	Su	Mo	Tu	We	Th	Fr	Sa	Su	Mo	Tu	We	Th	Fr	Sa	Su	Mo	Tu
Work	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Sprint 3	🟨	5	🟨	🟨	🟨	6	7	8	R																						
Kerfisstjórnun skilaverkefni 2																															
Ný Tækni: Assignment 2 due			🟩																												
Kerfisstjórnun skilaverkefni 3													🟩																		
Sprint 4									1	🟨	🟨	2	3	4	🟨	5	🟨	🟨	6	7	8	🟨	R								
Return for Status Meeting 2 1. Data according to Project plan 2. Progress Overview Report									R	L	🟥	🟥	🟥	🟥	🟥	🟥	🟥	🟥	🟥	🟥	🟥										
Status Meeting 2																SM	2	🟥	🟥	🟥	🟥										
Iðntölvur og Róbótar Verkefnavinna																🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
Hugbúnaðarfræði Verkefnavinna																🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
Kerfisstjórnun Verkefni 4																							🟩								
Sprint 5																							1	🟨	🟨	2	3	4	🟨	5	🟨
Ný Tækni assignment 3																															🟩
Iðntölvur og Róbótar Skilaverkefni 4																											🟩				

	April																													
	2												Exams														Week 13			
	We	Th	Fr	Sa	Su	Mo	Tu	We	Th	Fr	Sa	Su	Mo	Tu	We	Th	Fr	Sa	Su	Mo	Tu	We	Th	Fr	Sa	Su	Mo	Tu	We	Th
Work	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
Iðntölvur og Róbótar Verkefnavinna	🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟩	🟩	🟩			🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪						
Hugbúnaðarfræði Verkefnavinna	E	A	S	T	E	R	S						E	X	A	M	S	🟪	🟪	🟪	🟪	🟪	🟪	🟪						
Kerfisstjórnun Verkefni 4	E	A	S	T	E	R	S						E	X	A	M	S	🟪	🟪	🟪	🟪	🟪	🟪	🟪						
Sprint 5	🟨	6	7	8	🟨	R	🟪						🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪						
Ný Tækni assignment 3	🟪	🟪	🟪	🟪	🟪	🟪	🟪						🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪						
Iðntölvur og Róbótar Skilaverkefni 4	🟪	🟪	🟪	🟪	🟪	🟪	🟪						🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪						
Ný Tækni: Report due	🟪	🟪	🟪	🟪	🟪	🟪	🟪			R			🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪						
Kerfisstjórnun skilaverkefni 5	🟪	🟪	🟪	🟪	🟪	🟪	🟪			🟩			🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪						
Sprint 6	🟪	🟪	🟪	🟪	🟪	🟪	🟪						🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪			🟨	🟨	🟨	🟨
Return for Status Meeting 3 1. Administrators Manual 2. Users Manual 3. Data according to Project Plan	🟪	🟪	🟪	🟪	🟪	🟪	🟪						🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪					R	🟥

	May																		
				Week 14							Week 15								
	Fr	Sa	Su	Mo	Tu	We	Th	Fr	Sa	Su	Mo	Tu	We	Th	Fr	Sa	Su	Mo	Tu
Work	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
Sprint 6	🟨	🟨	🟨																
Return for Status Meeting 3 1. Administrators Manual 2. Users Manual 3. Data according to Project Plan	🟥	🟥	🟥	🟥															
Sprint 7				🟨	🟨	🟨	🟨	🟨	🟨	🟨									
Sprint 8											🟨	🟨	🟨	🟨	🟨	🟨	🟨		
Final Return of Project: Final Report Printed out All Data on CD or other Digital Media Code Included														FR					
Final Project Presentation																		🟥	🟥

Table 3: Time Plan Gantt Chart



Risk	Odds of risk factor	Loss of how many days	Days lost	Possible solution to a risk factor	Possible act to prevent risk factor	Person responsable for this issue
Changes of backlog and stories, estimate of how long it takes to finish a task	75%	12	9	Have a meating and redo the sprint estimation	Real time adjustments	Sveinn
Complexity of Valka´s previous code is grater than expected, lack of understanding	60%	8	4.8	Get a meating with one of current developers and do a code review	Ask lead programmer of Valk, and go through debuggin	Sveinn
Time spent testing is more than expected	60%	4	2.4	Find help libraries that makes the test enviroment easier to deal with	Time management and planning	Emil
Integrading CUDA in current Valka code is more compliated than expected	50%	4	2	Commit more time in CUDA integration	None	Emil
Sveinn's wife will have a baby before May 19	50%	4	2	Sveinn will work from home, Emil will work extra	None	Emil
Setting up scrum takes more time than expected	50%	3	1.5	Spend more time, look at slides from older courses	Look at older projects	Sveinn
The learning curve for CUDA is steeper than expected	45%	10	4.5	Research and spend time on google and Udacity online course, find somone who is familiar with CUDA paralell programming	Take the time to research, and go through Udacity program	Emil
Other courses are taking more time than estimated	40%	7	2.8	Organize the time better, skip other less important assignments	Organize,Organize,Organize	Sveinn
OpenCV is not compatible with current working setup	40%	2	0.8	Look at another libraries	Do research	Sveinn
The speed of the team is lower than estimated	30%	4	1.2	Do somthing diffirent for a small amount of time to get the moral up and commit more time	Evaluvate every week if the project is behind schedule	Sveinn
Testing new solutions to old results is delayed cause of coupled design in old code	25%	3	0.75	Implement functions in old code to generate data	Try to break down or debugg old code for some data points	Emil
Merging new master to branch takes more time to resolve than expected	25%	2	0.5	Merge branch with master often	Do this every time visiting Valka offices	Emil
Member of the team gets ill	20%	8	1.6	Call a doctor	Eat healthy and sleep enaugh	All team members
CUDA is performing slower than previous threaded code	20%	5	1	Take a step back and find a solution	Study the old algorithm, and setup a plan before implementation	Emil
Test machine is held up by other employees of Valka	10%	4	0.4	Ask Valka to get another test rig	No possible act	Sveinn
Test machine breaks down.	10%	2	0.2	Valka´s responsabiltiy to replace or get replacement parts	The computer is brand new, do not install unknown software	Sveinn
Members computer breaks down	10%	2	0.2	Borrow a replacement	None	All team members
Member breaks the confidentialy agreement and Valka stops development	2%	40	0.8	Find a new team member or finish on your own	Keep confidentiality	Sveinn
Team member leaves the team	0.5%	40	0.2	Find a new team member or finish on your own	Keep the moral positive	All team members
Sum of days lost						**36.65**

Color Codes
- 🟨 Risk factor occured
- 🟩 New risk factor
- 🟦 Obsolete

Table 4: Risk Analysis

6.1 Product Backlog

The product backlog undertook some changed through out the span of the project. The biggest change in the backlog can be seen in chapter 6.4 where the team created a new test framework. No other major changes were made to the backlog.

6.2 Sprint 0

Sprint 0 was used for preparation and project setup. The main goal of this sprint was to have a development environment setup as well as creating project management spreadsheet. The team succeeded their goals. To accelerate understanding of how the CUDA library works the members of the team turned to `http://udacity.com` a online courses on parallel programming. Udacity provided the necessary knowledge with excellent explanation to get the the basics of CUDA. The sprint burndown can be seen in figure 3.

6.2.1 Stories completed

- *As an instructor I can read first draft of work schedule so I can grade the team:* Story points: 8.
- *As an Instructor I can read risk management so I can grade the team and evaluate risks:* Story points: 2.
- *As a developer I can build code in visual studio so I can use the project:* Story points 2.

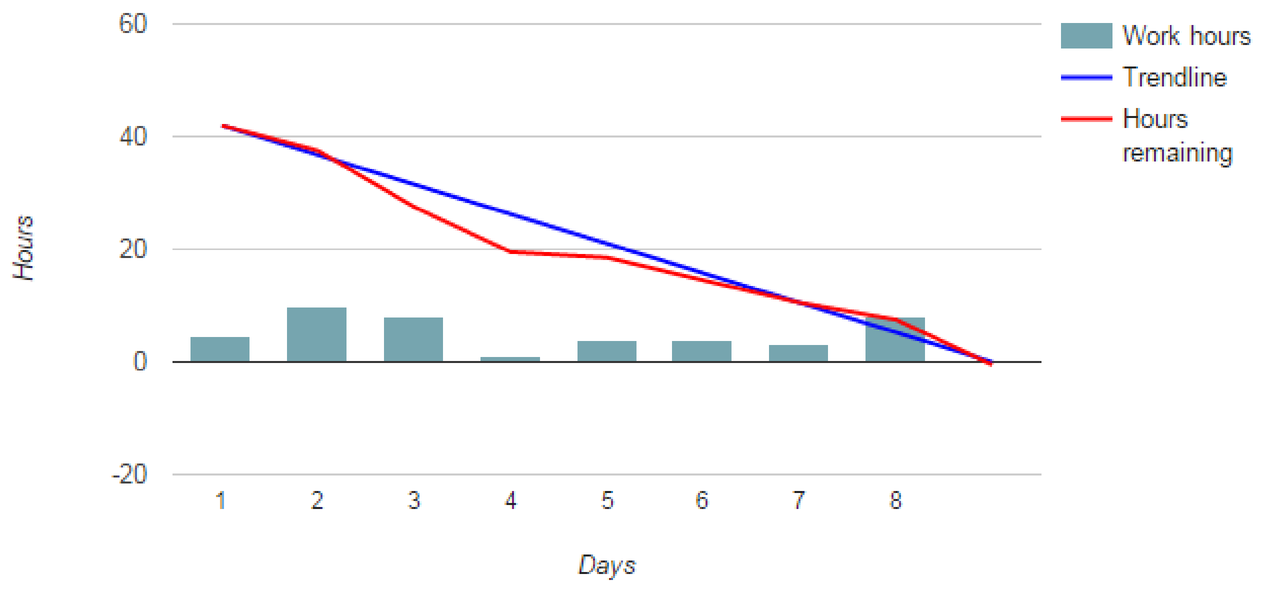


Figure 3: Sprint 0 Burndown Chart

- *As a developer I can use my own branch in subversion so my work does not interfere with other developers:* Story points: 1.
- *As a developer I have to take the Udacity online course so I can program in CUDA:* Story points: 16.

Total hours: 45.

6.2.2 What went well

The sprint was a success. Everything worked out as planned.

6.2.3 What could have gone better

Everything worked out as planned.

6.2.4 Possible improvements

No improvements needed.

6.3 Sprint 1

In sprint 1 the goal was to be able to run CUDA kernels inside Valka X-ray Guided Cutting Machine's code base. The team accomplished the goal. Most of the work hours were spent writing a report for the first status meeting. Figure 4 shows the burndown chart for sprint 1.

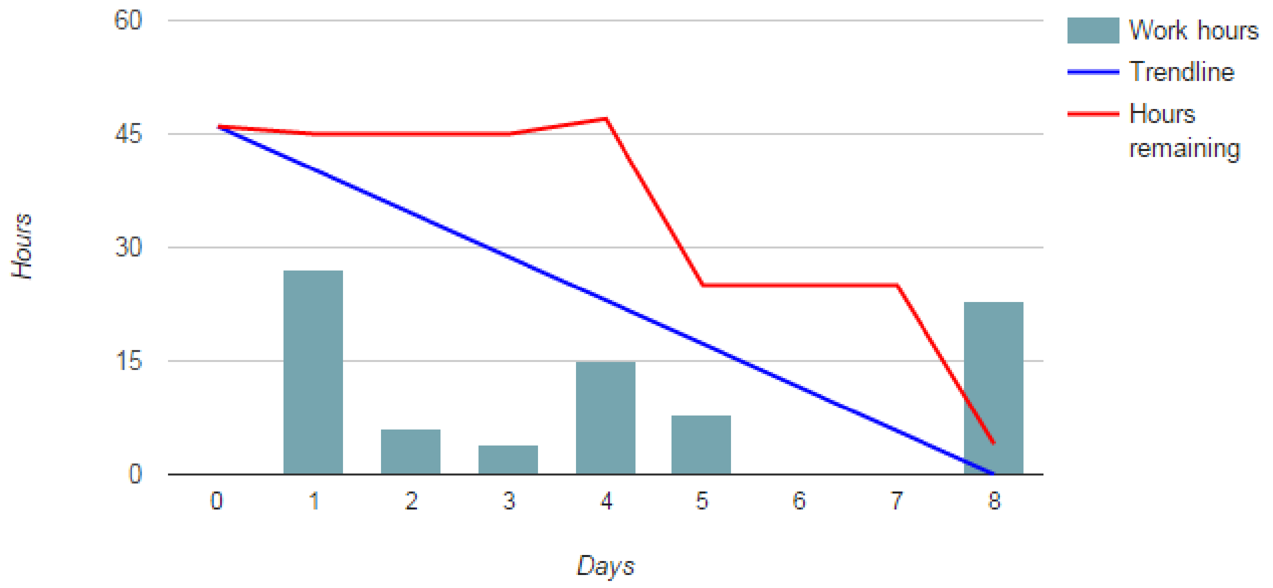


Figure 4: Sprint 1 Burndown Chart

6.3.1 Stories completed

- *As an instructor I can read the report before first status meeting:* Story points: 8.
- *As an instructor I can listen to the developer presentation on first status meeting:* Story points: 8.
- *As a co student I have been presented the project:* Story points: 4.
- *As a developer I can add CUDA framework/lib to the solution so CUDA code will build:* Story points: 2
- *As a developer I can benchmark CUDA version of function to old version of function so I can provide data to the report:* Story points: 8.

Total hours: 91.

6.3.2 What went well

The sprint was a success. Everything worked out as planned.

6.3.3 What could have gone better

Everything worked out as planned.

6.3.4 Possible improvements

No improvements needed.

6.4 Sprint 2

The team encountered obstacles in the original plan to modify current Valka code to CUDA. The first was the unforeseen CUDA warmup time of 300ms for each instance calling CUDA. With current design of the Valka X-ray Guided Cutting Machine's codebase, it would mean 300ms overhead for bone detection of every fillet which is is not acceptable. The second obstacle was that complexity and tight coupling of current codebase slowed down the design and implementation work of the CUDA version of the code. For more information of CUDA warmup see:
`http://forums.udacity.com/questions/100045265/warming-up-effect-in-gpu-distorts-the-comparison-of-times`.

Those obstacles forced the team to make slight changes to the original plan. Instead of modifying current bone detection library, the team opted to redesign the code and designed a simpler test framework. This was done after careful consideration and in collaboration with Helgi Hjálmarsson at Valka and Stefán Freyr Stefánsson, the teams instructor. The sprint undertook some changes because of those obstacles as well did the product backlog. As seen in figure 5 a considerable amount of stories were removed at the end of this sprint because of those complications.

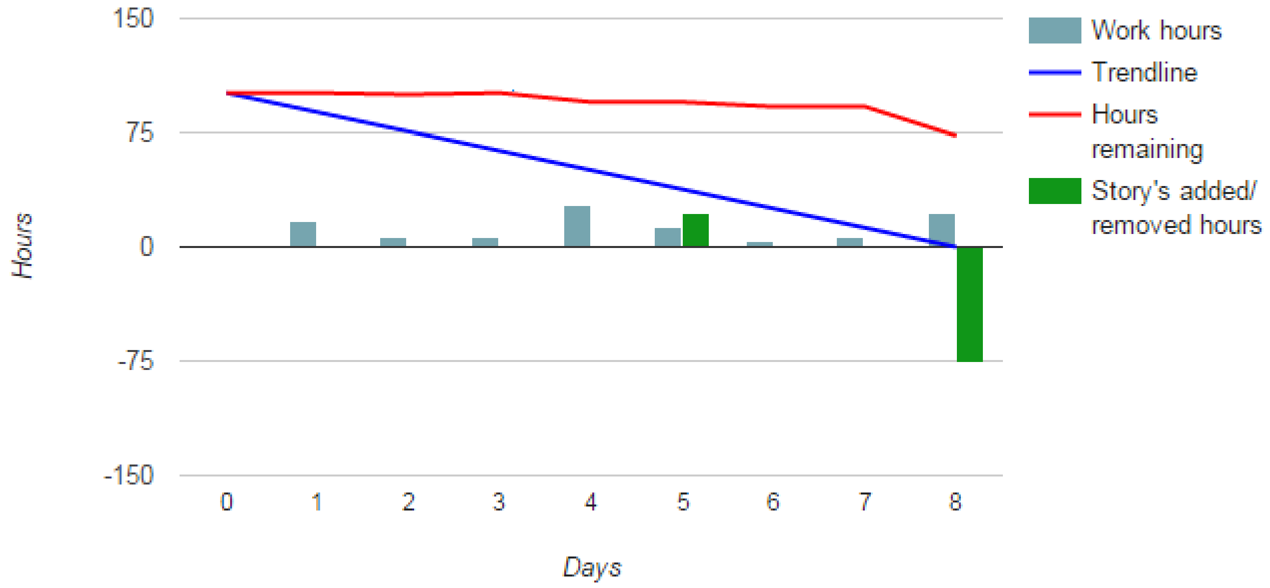


Figure 5: Sprint 2 Burndown Chart

6.4.1 Stories completed

- *Merge and update to newest version of the code:* Story points: 4.
- *Make new smaller project with new API design:* Added to sprint. Story points : 4.

- *As Valka Cutter I can use CUDA version of methodA() so I can work more efficiently:* Added to sprint. Story points: 8.

6.4.2 Stories not completed

- *As Valka Cutter I can use CUDA version of methodE() so I can work more efficiently:* Moved to sprint 3.
- *As Valka Cutter I can use CUDA version of methodF() so I can work more efficiently:* Started work on the story. Moved to sprint 3.
- *As Valka Cutter I can use CUDA version of methodG() so I can work more efficiently:* Moved to sprint 3.
- *As Valka Cutter I can use CUDA version of methodK() so I can work more efficiently:* Moved to sprint 3.

Total hours: 101.

6.4.3 What went well

The team reacted to the obstacles it faced by making a slight change in direction of the project without affecting the end result. Risk management plan worked out as it should and the team did not fall far off its course.

6.4.4 What could have gone better

Although the team encountered obstacles the team's reactions held the team on course.

6.4.5 Possible improvements

No improvements needed.

6.5 Sprint 3

After the redesign made in sprint 2 the team started working on the CUDA version of the new bone detection library. The sprint progressed better than anticipated and the team had to add new stories twice, total of 28 story points. Even though the risk of increased workload in other courses occurred as discussed in chapter 5, the team delivered. Figure 6 shows the sprint burndown. The team finished all the preprocess methods and were well on way with the analyze methods. A total of 62 story points were completed.

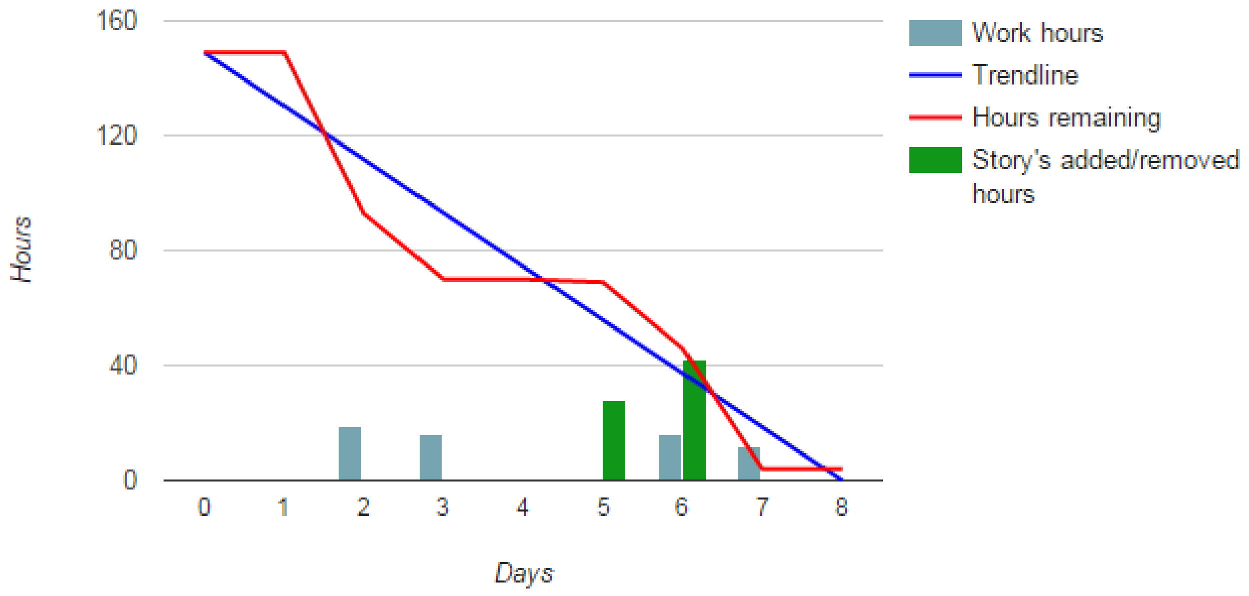


Figure 6: Sprint 3 Burndown Chart

6.5.1 Stories completed

- *As Valka Cutter I can use CUDA version of methodB() so I can work more efficiently:* Story points: 8.
- *As Valka Cutter I can use CUDA version of methodB1() so I can work more efficiently:* Story points: 8.
- *As Valka Cutter I can use CUDA version of methodC1() so I can work more efficiently:* Story points: 4.
- *As Valka Cutter I can use CUDA version of methodC2() so I can work more efficiently:* Story points: 4.
- *As Valka Cutter I can use CUDA version of methodC() so I can work more efficiently:* Story points: 8.
- *As Valka Cutter I can use CUDA version of methodD() so I can work more efficiently:* Story points: 2.
- *As Valka Cutter I can use CUDA version of methodE() so I can work more efficiently:* Added to sprint. Story points: 2.
- *As Valka Cutter I can use CUDA version of methodE1() so I can work more efficiently:* Added to sprint. Story points: 4.
- *As Valka Cutter I can use CUDA version of methodE2() so I can work more efficiently* Added to sprint. Story points: 2.

- *As Valka Cutter I can use CUDA version of methodP() so I can work more efficiently:* Added to sprint. Story points: 4.
- *As Valka Cutter I can use CUDA version of methodF() so I can work more efficiently:* Added to sprint. Story points: 8.
- *As Valka Cutter I can use CUDA version of methodG() so I can work more efficiently:* Added to sprint. Story points: 8.

Total hours: 63.

6.5.2 What went well

The sprint was a success. Stories were finished faster than anticipated and new badge of stories were added twice, total of six stories.

6.5.3 What could have gone better

Everything worked out very well.

6.5.4 Possible improvements

No improvements needed.

6.6 Sprint 4

In sprint 4 the team only managed to finish 36 story points. This is due to time spent on status report, presentation for status meeting 2 and time spent figuring out how to implement reduction in CUDA. Eventually the team found an acceptable solution for reduction at `https://a248.e.akamai.net/f/862/5658/3/developer.download.nvidia.com/compute/cuda/1.1-Beta/x86_website/projects/reduction/doc/reduction.pdf`. This solution was modified and put to use in *methodH()* and *methodI()* functions. Burndown chart for sprint 4 can be seen in figure 7.

6.6.1 Stories completed

- *As an instructor I can read the report before status meeting 2:* Story points: 8.
- *As an instructor I can listen to the developer presentation on status meeting 2:* Story points: 4.
- *As Valka Cutter I can use CUDA version of methodH() so I can work more efficiently:* Story points: 8.

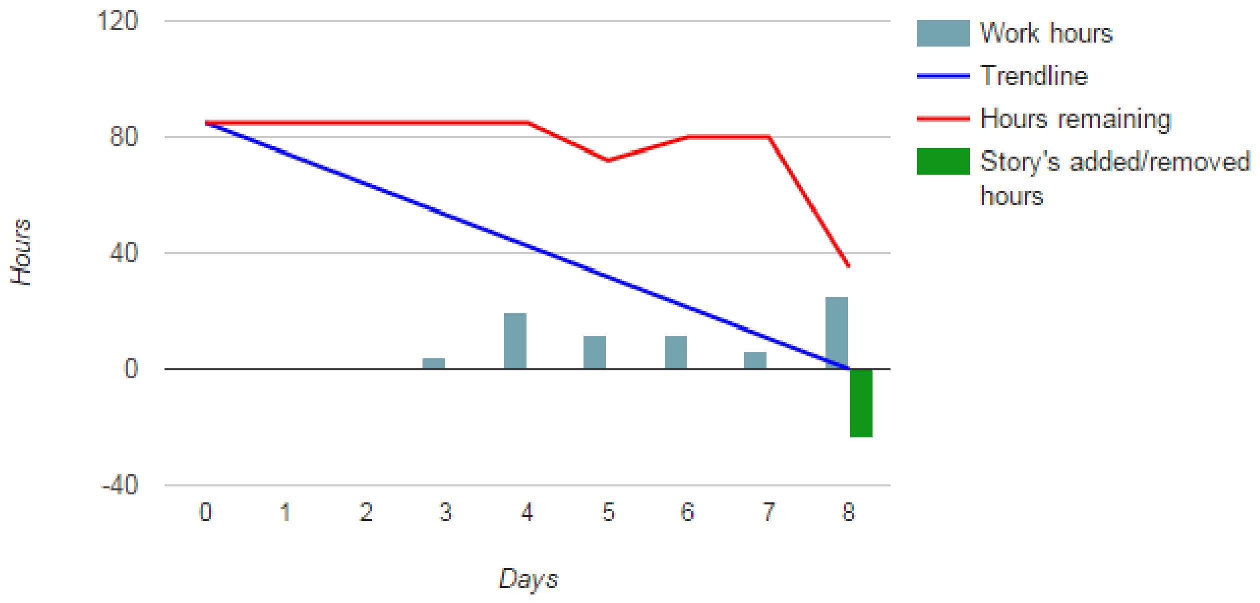


Figure 7: Sprint 4 Burndown Chart

- *As Valka Cutter I can use CUDA version of methodI() so I can work more efficiently:* Story points: 8.
- *As Valka Cutter I can use CUDA version of methodJ() so I can work more efficiently:* Story points: 8.

6.6.2 Stories not completed

- *As Valka Cutter I can use CUDA version of methodK1() so I can work more efficiently:* Removed from sprint.
- *As Valka Cutter I can use CUDA version of methodK() so I can work more efficiently:* Removed from sprint
- *As Valka Cutter I can use CUDA version of methodL() so I can work more efficiently:* Removed from sprint.

Total hours: 79.

6.6.3 What went well

The team solved the reduction problem.

6.6.4 What could have gone better

The time spent figuring out the reduction problem was greater than anticipated, therefore three stories were moved to sprint 5.

6.6.5 Possible improvements

Better evaluation of story points.

6.7 Sprint 5

The goal for sprint 5 was to have a working prototype of the new CUDA bone detection software. Progress was grater than anticipated and succeeded the goal. The team finished all coding and had a working prototype. This prototype was around two times faster than the original version without any CPU load. Increased CPU load was likely to make the original version somewhat slower, which will increase the speed gap between the CUDA version and the original CPU version. Figure 8 shows how sprint 5 progressed. As can be seen the sprint finished in the first four days, which was well appreciated by the team since workload in other courses had increased as the semester was coming to an end.

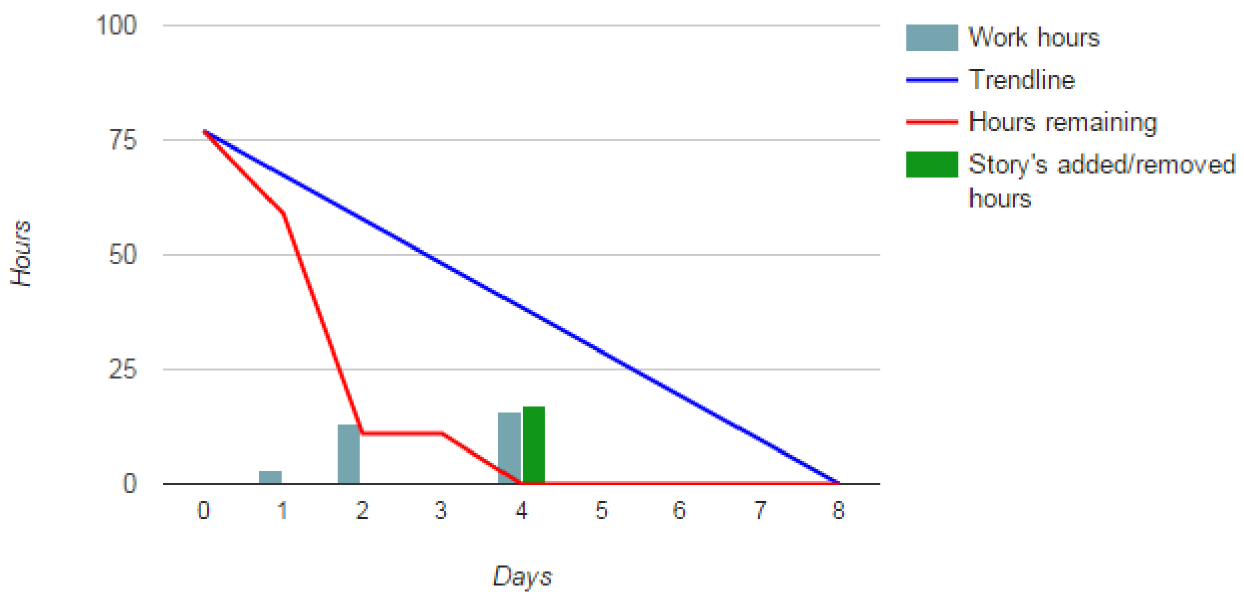


Figure 8: Sprint 5 Burndown Chart

6.7.1 Stories completed

- *As Valka Cutter I can use CUDA version of methodK1() so I can work more efficiently:* Story points: 8.
- *As Valka Cutter I can use CUDA version of methodK() so I can work more efficiently:* Story points: 8.
- *As Valka Cutter I can use CUDA version of methodL() so I can work more efficiently:* Story points: 8.

- *As Valka Cutter I can use CUDA version of methodM() so I can work more efficiently:* Story points: 4.
- *As Valka Cutter I can use CUDA version of methodN() so I can work more efficiently:* Story points: 4.
- *As Valka Cutter I can use CUDA version of methodO() so I can work more efficiently:* Story points: 8.
- *Fix methodF(), methodG() and methodO(), match output to original images:* Story points: 4.
- *As Valka Cutter I can use CUDA version of methodR() so I can work more efficiently:* Story points: 4.
- *Fix methodF() and methodG() and methodO():* Added to sprint. Story points: 8.
- *As Valka Cutter I can use CUDA version of methodR() so I can work more efficiently:* Added to sprint. Story points: 4.

Total hours: 32.

6.7.2 What went well

The team reached their goal to have a working prototype of the XFBCUDA library sooner than anticipated, since the last two stories to achieve the goal were added on day 4. The team worked fast and efficiently.

6.7.3 What could have gone better

Everything worked out as planned.

6.7.4 Possible improvements

No improvements needed.

6.8 Sprint 6

The main objectives in sprint 6 was firstly to review the code, refactor, standardize documentations and recheck all memory allocations. Secondly timing and benchmarking the new CUDA version to the existing CPU version. Lastly to be well on the way of writing the final report and the administration manual.

The team successfully finished all refactoring and declared a code freeze in the XFBCUDA library. Work on the final report went according to plan. Burndown chart for sprint 6 can be seen in figure 9.

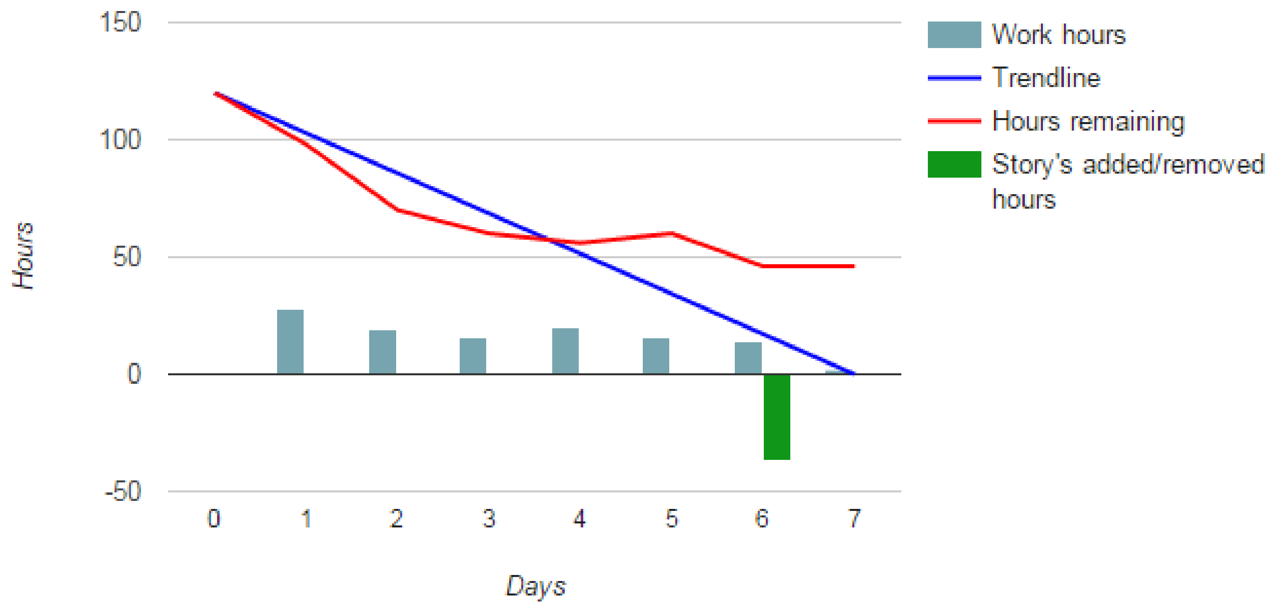


Figure 9: Sprint 6 Burndown Chart

6.8.1 Stories completed

- *Re-check delete and free memory, use N-sigth profiler to verify:* Story points: 4
- *Code refactoring (unused variables and more):* Story points: 8.
- *Doxygen for code:* Story points: 8.

6.8.2 Stories not completed

- *Final Report:* Draft finished. Moved to sprint 7.
- *Final Presentation:* Not started. Moved to sprint 7.
- *Administration manual:* Mostly finished. Moved to sprint 7.
- *User manual:* Merged with administrator manual.

Total hours: 113.

6.8.3 What went well

The team succeeded in their goals.

6.8.4 What could have gone better

Everything worked out as planned.

6.8.5 Possible improvements

No improvements needed.

6.9 Sprint 7

The main goal of sprint 7 was to have compared OpenCV methods to Valka methods and see if OpenCV is a viable option for further development of the Valka software. Also the team needed to finish the report and presentation for status meeting 3. Both goals were accomplished although installation of the OpenCV for Visual Studio 2010 was more complicated than anticipated. Figure 10 shows the sprint burndown.

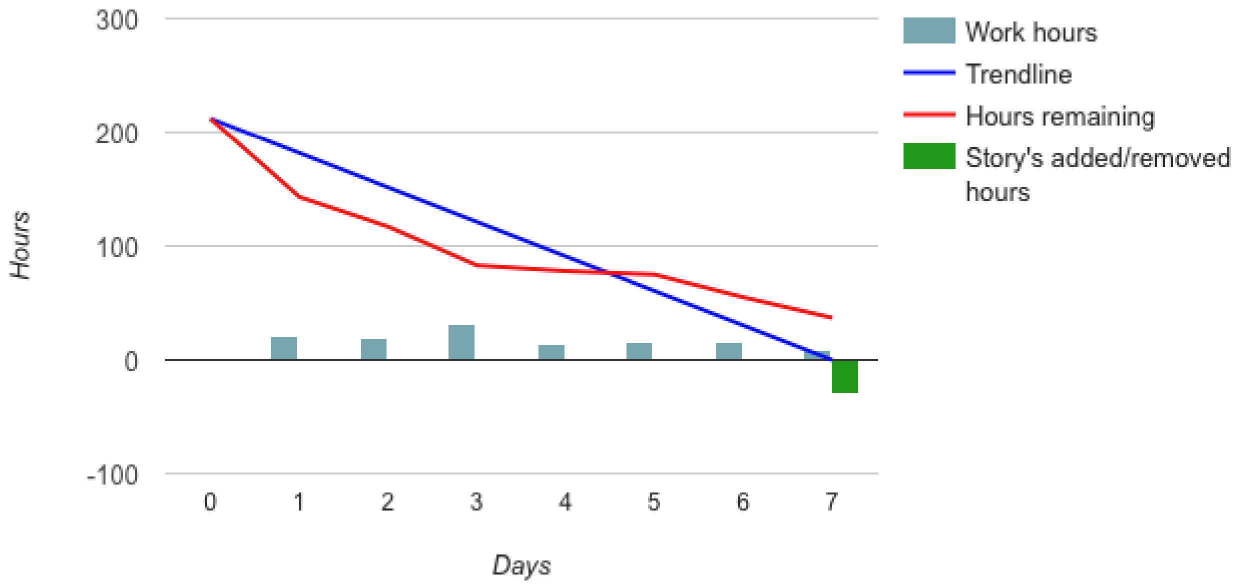


Figure 10: Sprint 7 Burndown Chart

6.9.1 Stories completed

- *As a developer I can access OpenCV on the branch:* Story points: 2.
- *As a developer I can use OpenCV on the Dev machine so I can code solutions in OpenCV:* Story points: 16.
- *OpenCv::GPU implementation of methodG():* Story points: 8.
- *OpenCV implementation of methodG():* Story points: 8.

- *Report for Status Meeting 3:* Story points: 16.
- *Presentation for Status meeting 3:* Story points: 16.
- *Administration manual:* Story points: 8.

6.9.2 Stories not completed

- *Final Report:* Not finished. Moved to sprint 8.
- *Final Presentation:* Not started. Moved to sprint 8.

Total hours: 125,5.

6.9.3 What went well

The team finished benchmarking OpenCV and went far with the final report.

6.9.4 What could have gone better

OpenCV library was harder to install than anticipated.

6.9.5 Possible improvements

No need for improvements.

6.10 Sprint 8

Sprint 8 was only three days and was used to finish the final report and the final presentation. Figure 10 shows the sprint burndown.

6.10.1 Stories completed

- *Final Report:* Story points 8.
- *Final Presentation:* Story points.

Total hours: 46

6.10.2 What went well

The sprint was a success. Everything worked out as planned.

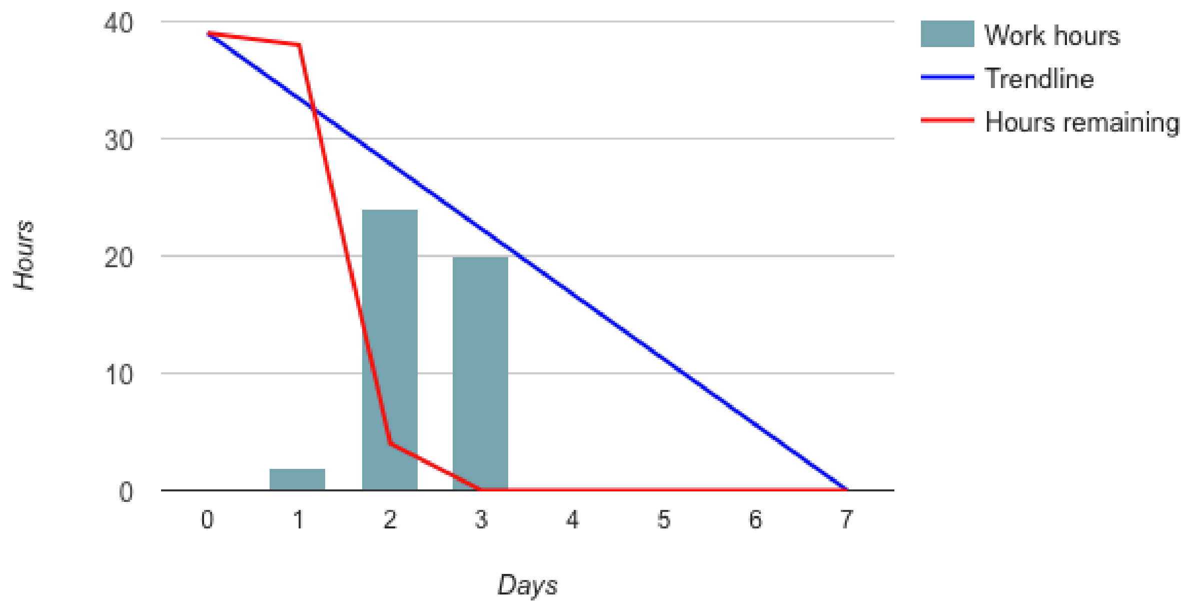


Figure 11: Sprint 8 Burndown Chart

6.10.3 What could have gone better

Everything worked out as planned.

6.10.4 Possible improvements

No improvements needed.

6.11 Release

At the end of sprint 8 the XFBCUDA library is fully functional and ready to include in the Valka Cutting machine. Benchmarking has been done and OpenCV observed. The product backlog contained 383 points and the team held course the whole time. Although difficulties in sprint 2 and sprint 4 slowed the team down a bit, they delivered and finished the project on time. The product burndown can be seen in figure 12. The team's velocity increased with time, 0,5 point/hour in the beginning and nearly 1,0 point/hour according to the trend line of the team's velocity chart shown in figure 13.

6.12 Time Management

A total of 690 hours was spent on the project. 46% of the time was spent on programming, 25% on reports, 14% on presentations and 10% on scrum related work. The rest was spent on lectures and data gathering. Figure 14 shows time management.

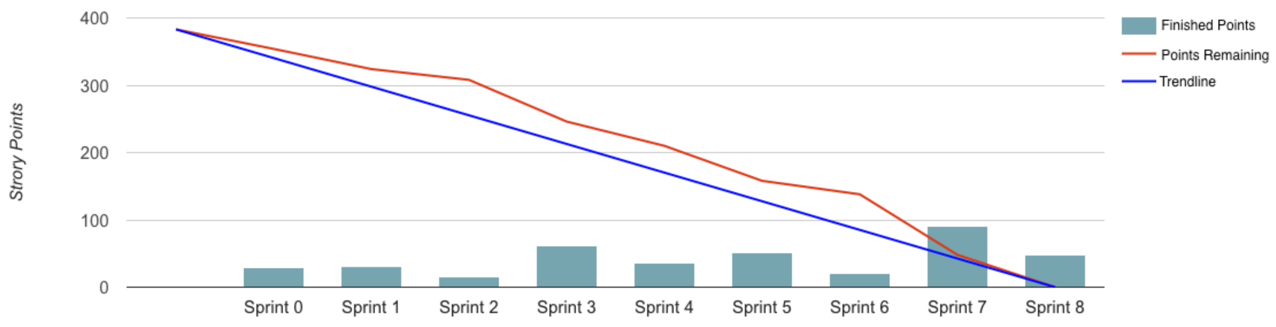


Figure 12: Release Burndown Chart

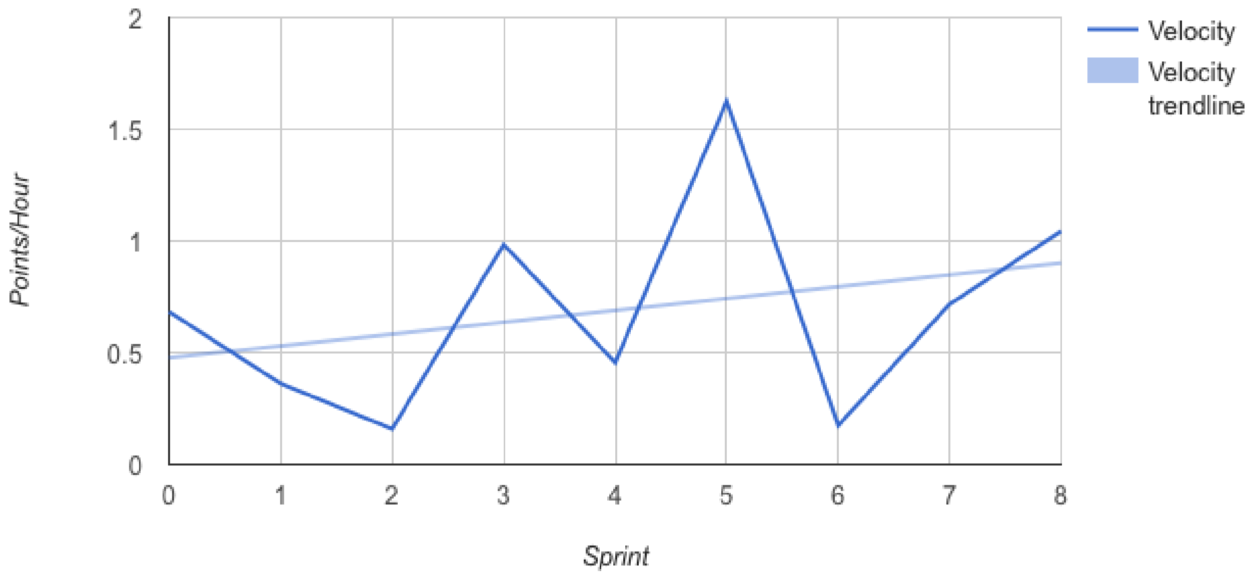


Figure 13: Team Velocity Chart

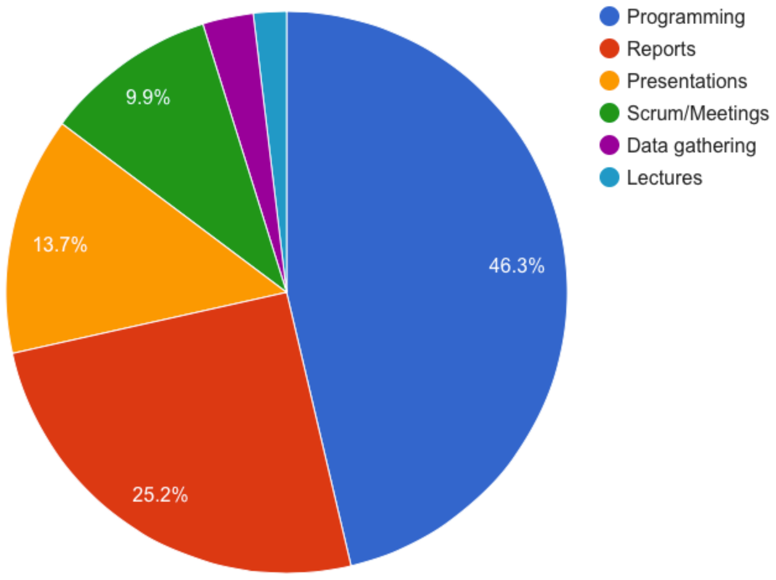


Figure 14: Team Work Time Management Chart

7 Results and Discussion

CUDA implementation of the bone detection is 2,55 times faster than the original CPU version. Given 1% increase in throughput when processing 6.000 metric tons of cod a year, could result in increased yearly revenues of €480.000. Integration of the GPU implementation into Valka's current code base could highly benefit the fish industry.

Since the team had no experience with CUDA when starting the project the programming part was uphill battle to begin with. Previous knowledge from courses like Operating systems and Algorithms came to good use.

Many algorithms can benefit of being ported to CUDA but there are also cases where a CPU does better job. During the development of this project the team hit some of the CUDA weak points. When it comes to sorting and summing a dataset CUDA can become less effective although with the right method, good results can be achieved.

The speed improvements are most obvious in the methodF() function, where it was possible to exploit CUDA's strengths to the maximum. The CUDA version of methodF() runs over seven times faster than the original one, which is around 230ms faster than the CPU version.

During the development of methodH(), as discussed in chapter 6.6, which uses a sum function for calculation, the disadvantages of a naive CUDA implementation is clear. Using basic atomic add method would require a lot of thread locking and result in a slower performance than CPU version would otherwise do because of thread serialization. The naive atomic add implementation of the function resulted in 10 times slower execution time than the CPU implementation.

The solution was to implement sum reduction where each block of threads calculates the sum of itself and writes the result to partial sum. When all of the blocks have completed their local sum, the same kernel is execute once more with the partial sum as input, upon completion index[0] of the output array contains the sum of the image. The result show the execution time of the CPU vs GPU are similar. Even though the execution time is similar, it is worth mentioning the CPU version uses OpenMP reduction as well. Without OpenMP the CUDA version would be considerably faster than the CPU version.

Some of the functions use similar methods. Those functions usually perform a single task on each pixel. These tasks are simple, such as checking threshold or multiplication of pixels. The functions share a similar time improvement, around two times faster than the

CPU version using OpenMP. To put this into perspective, if a marathon runner is two times faster than its competitor, the winner is at the finish line when the competitor is only half way there.

It is worth noticing, the CPU version of the Valka bone detection software is highly optimized. It uses OpenMP, where tasks are executed in parallel along with other optimizing methods so the team had to work hard to improve the speed of the original CPU version.

The OpenCV library contains various filters, using the filters can speed up the development process. On the other hand trying to port a highly specified filter method as used in Valka bone detection is next to impossible or would require rewriting some of the already implemented filters. Nevertheless the filters available in OpenCV are fast and can suit the bone detection, therefore OpenCV would make a strong candidate if the need for new image processing functionality or rewriting the bone detection from scratch would occur.

8 Conclusion

The project went through small changes as it progressed. The original plan to take the whole Valka Cutter project and modify the functions to use CUDA was not possible as reported in chapter 6.4. The fast recognition of this problem made it easy to adopt and the solution was to create a new simple test project. This accelerated the development process and made it possible to finish both the profile image analysis and the bone detection, which was the main objective of the project. The lower priortiy objective of the project was to investigate if OpenCV is a viable option for Valka to use in future development. As seen in chapter 7, OpenCV is a good asset to have when dealing with image processing and mathematical problems based on matrix calculations.

In the beginning it was a known fact, in order to be able to show some execution time improvement, the methodF() would play a big role. Execution time improvement of the methodF() is over seven times faster than the original CPU version. Taking the original execution time of methodF() down from 270ms to only 36,7ms was appreciated by the team and showed that CUDA has a future in the Valka X-ray Guided Cutting Machine and other Valka projects.

Overall execution time of the CUDA version of the bone detection is 137ms which is half the time of the original CPU methodF() function and 2,55 times faster than the total time of the existing CPU version. As stated in chapter 7, a marathon runner would be pretty happy finishing the race when the next runner has only half way to the finish line.

Acknowledgments

Reykjavík University: Stefán Freyr Stefánsson, Torfi Helgi Leifsson and Hallgrímur Arnalds for commenting and reviewing the project.
Valka: Helgi Hjálmarsson, Einar Björn Jónsson, Guðmundur Freyr Aðalssteinsson, Helga Ingimundardóttir and Hannes K. Gunnarsson for explanation of the Valka code and other useful information.
Wifes and girlfriends: Ellen Ellertsdóttir for reviewing and patience and María Ósk Stefánsdóttir for patience.

References

[1] Valka, “Valka website,” apr 2015. [Online]. Available: http://valka.is/

[2] NVIDIA, “NVIDIA Website,” apr 2015. [Online]. Available: http://www.nvidia.com/object/cuda_home_new.html

[3] Itseez, “Itseez website,” apr 2015. [Online]. Available: http://itseez.com/OpenCV/