Adam Kurkiewicz

Github Projects

Predictive Collision

A priori collision engine in Python

Quick and accurate collision detection is a big algorithmic challenge the game industry faces every day. Most solutions use *a posteriori* approach, which requires a series of calculations to be performed every frame. In this project I have explored a different approach, which computes the exact position of the collision event using linear algebra. The advantage lies in calculations being performed only once every 10-100 frames, and disadvantage is their higher computational cost.

view on github →

Vaida

Humanizing public-key cryptography

Vaida is a solution, which redefines decentralized key distribution. With simple user interface and a secure protocol, Vaida lets the user create a video with personalized statement, which contains their key properties. Subsequently the statement is made public, e.g. on youtube, and the user's peers can verify user's identity and their new public key. *Expected release date January 2014*

view on github →

Stock Exchange

Scalable stock exchange in Java

This is a team effort undertaken during Barclay's hackathon in Autumn 2013. Along with 3 other participants I have managed to create a scalable stock exchange, capable of executing 300 000 trades per second. The code shows a genuine 24-hour effort, and no post-hackathon development was carried out. I have plans to carry on with the project, depending on other commitments.

view on github →

Strongly Connected Divide and Conquer

Parallel implementation

Perhaps the finest of my programmatic achievements, this work in Java is an implementation of an algorithm introduced in the following IBM research report. The algorithm performs a decomposition of a graph into Strongly Connected Components – a very important graph operation, which is frequently the first step in analysis of large graphs. Prior to early 2000s, we had not known efficient algorithms to perform SCC decomposition in parallel, and although we do now, there is a shortage of their implementations. The following work is my attempt to bridge this gap.

view on github →

✍ 11 Clarence Gardens
G11 7JN Glasgow
☎ +44 07510 554340
✉ adam@kurkiewicz.pl
● www.github.com/picrin

Education

2011 – 2015	University of Glasgow **JH Maths & Computing** *Glasgow, UK*
2010 – 2011	Jagiellonian University **Studies in science** *Krakow, Poland*
2007 – 2010	Melchior Wankowicz HS **International Baccalaureate** *Katowice, Poland*
2004 – 2007	French-Polish bilingual JH **Certificate in French** *Katowice, Poland*
1998 – 2004	Music Primary School **Main instrument: Cello** *Katowice, Poland*

Skillset

Organiser

In May 2013 I have co-founded GUTS – Glasgow University Tech Society. Currently I serve as a projects officer in the society and a leader of scientific programming group. My biggest achievement within society was co-organisation of a hackathon, which has attracted about 50 participants and over a dozen of corporate sponsors.

Team player

I am a firm believer of pair programming. Both my coursework and non-academic projects often involve this technique. Especially when my partner is more fluent in a specific technology, pair programming is the quickest way I can catch up with. As a part of the team I always try to serve the best of my knowledge, which usually means taking the responsibility for picking correct algorithms and data structures, reasoning about the correctness of programs, writing critical sections of parallel programs and implementing secure parsers. The above area of expertise stems from my interest in mathematics, which is a part of my degree.

Confident

Whenever posed with a technical challenge, I approach it with can-do attitude. The first step I apply is trying to find an existing, possibly open-source technology, which would largely facilitate or even solve the problem. If writing the code is a necessity, I like to split up the work into small, manageable tasks. Every little achievement keeps the spirits high and makes the workload feel less overwhelming.

Practical

Although excited by new technologies, I would not push to rewrite entire legacy system in the cool new language of the day. I always try to pick the best tool for the job and try not to discard particular technology only because I am not fond of it. In the end of the day, all what matters is good quality code. And good quality code is an attribute of the programmer, not of the language.

Toolset

Python

I like to use Python the way it likes to be used – exploiting its highly dynamic structure. Runtime method definition makes it easy to write powerful loggers, mixins help to keep implementation of separate features separate, and decorators are a nice way of introducing helpful functional paradigm features into OOP. I like to use python-flask for web devolopment, matplotlib for automatic graph generation, python-gdata for scripting google forms and pygame for more advanced visualisations and games.

Java

I like to use Java for computationally intensive tasks, for powerful parallelism and server-side programming. I am a fan of JVM languages family, currently learning scala, which I believe is going to be a successor to Java.

Functional languages

Although I can't claim proficiency in any of those, I hope to put a much larger stress on the functional paradigm in my future development. Recently I have been exploring happstack – haskell's web-server.

Platform

Linux is my OS of choice. I love package managers and the freedom of experimenting with different distros and window managers. I am an experienced git user, I am familiar with bash scripting, and I can set up a LAMP stack. I can trouble shoot both software and hardware problems. I use mostly LaTeX for typesetting my documents, but I am familiar with libre-office and google-docs. I have done simple scripting of .xls spreadsheets and am able to work with MS Office suit.