|  |
| --- |
| Maths and computer graphics Coursework 2 |
| Approximation of Inverse Kinematics using Dual Quaternions |
| An implementation of a robot arm within Octet |
|  |
| Juanmi Huertas Delgado, Sam Hayhurst and Sokol Murturi |
| 1/22/2015 |

|  |
| --- |
| Keywords: Quaternions, Dual-Quaternions, Inverse Kinematics, Interpolation, Raycasting, clostest neighbour approximation. |

# Introduction

Sox can you do this section please mate, make it maybe 1/2 to 3/4 of the page. We'll put a picture in to cover the rest of the page.

* Why did we choose the project?
* What was the aim of the project?

# Dual-Quaternion Implementation

Juanmi this is your section, remember to include references to the papers we read, used and copied.  
some pseudo code here would be forking awesome.

# Robot Arm and Inverse Kinematics approximation overview

Sam this is your section, this section will be picture heavy and discuss how we didnt attempt IK and how we currently do it. also why we chose this method od showing off our dual-quaternion class.

## Robot Arm

The decision to visualise the work using a robot arm was due to a number factors. The first being robot arms were the subject of many of the papers we looked at when searching for inverse kinematics and quaternions. Secondly, we are game developers; we wanted more than just a technical demo to show off on our portfolio.

To build the robot arm we built a base class called DQ\_Bone which was, unsurprisingly, made to represent a skeleton bone. the class contained dual quaternions which were used to orient the bone itself. The class also contains the joint to which it is attached, this is so that if a constraint is set on the joint, it affects the orientation of the bone to which the joint is a member. For example: the elbow constraint affects the orientation of the forearm.

Each bone, apart from the root, has a pointer to its parent and child. These allow other parts of the game to access the world versions of the dual-quaternions, specifically the animation and the magnetism. We introduced the magnetism force such the player (or user) would be forced to interact with both the inverse kinematics and the animation of the arm.

## Inverse Kinematics

Our original idea was to use traditional Inverse Kinematics to move the arm, coupled with ray casting this would allow the user to click a point in the world which the arm would then slerp towards. Due to time constraints and the likelihood of application we instead, per Andy's advice, implemented a closest neighbour seeking algorithm to animate the skeleton. This solution allowed for animation to easily and quickly be implemented, with a fast enough step and rotational constraints on the joints the animation is smooth and relatively natural.

The natural motion of the robot arm, despite the stochastic nature of the animation, is due to nature of quaternions themselves. The fact that linearly interpolating quaternions results in the shortest path meant that the animation is naturally smooth.

# User Interaction

Sam again, how does the player interact with the system. include bits about gaming and ray tracing

To

# What We Learnt

all should have input here, what did we learn as a collective on this project.

# Conclusions, Discussion & Further Work

last thing to do guys then we git push this doc

# References

fill them up buttercup I want Stanford - Imperial referencing ;D