Project Documentation - Quaternion Cipher

Author: Sean Oreta
CWID: 886349166

For my project, I attempted to utilize a concept I learned in my Computer Graphics class to create a symmetric key cipher that uses Quaternions to rotate plaintext, similarly to a rotation cipher but in a 3D plane.

Quaternions

Quaternions are a number system extending from complex numbers which are widely used in transforming and rotating 3D objects. This is defined as:
q=w+xi+yj+zk

where q is the scalar component, x, y, and z are the vector components, and i, j, and k are the imaginary units which follow the rule:

i^2=j^2=k^2=ijk=-1

Being used extensively in applications like Computer Graphics and Robotics, this system forms the basis to scramble text in a 3D space.

Algorithm

The intended operation of this cipher can be summarized as the following:

1. Each character of the plaintext is transposed onto a set of 3x3 vectors, each representing the points of a cube in a 3d plane. Extra space is padded out with a special '\x1F' char.
2. Using Quaternion values defined by a provided key, the program performs a set of rotations on each cube, jumbling the plaintext.
3. In order to decrypt, the program goes through the key's rotations in reverse by inverting the Quaternions.

Key generation is also fairly simple, as it generates a pattern of Quaternions that are followed for both encryption and decryption. These Quaternions are chosen at random from a pregenerated list, as allowing for the full range of rotations may move characters into awkward positions, such as between points. The user is freely able to choose the key size.

Results & Limitations

Limitations

The cipher itself runs fine, though there are a few problems with the implementation. As I could only integrate a few types of Quaternion rotations, the cipher is susceptible to brute force attacks from those familiar with how they work. Since the Quaternion rotations are applied uniformly thought the entire cube, it creates a pattern that could be exploited in the event of an attack. Additionally, while it works quickly in smaller test cases, the algorithm itself is not very fast on larger plaintext.

Further Improvements

If given the chance to further develop this cipher, implementing a less uniform rotation algorithm that functions similarly to a Rubik's Cube would allow the rotations to have less of a pattern and greatly increase the potential combinations. Improving key generation so that the Quaternions aren't stored as plaintext floats would also be a major improvement to prevent onlookers from easily surmising the context. Furthermore, changing to an asymmetric key model would add security to the cipher.

Acknowledgements

I would like to thank my Computer Graphics professor for introducing me to the concept of Quaternions, as well as the various online resources that helped me understand the concept enough to implement it in this project.
I would also like to acknowledge ChatGPT for its assistance in prototyping and refining both the source code and documentation for this project.