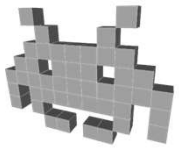
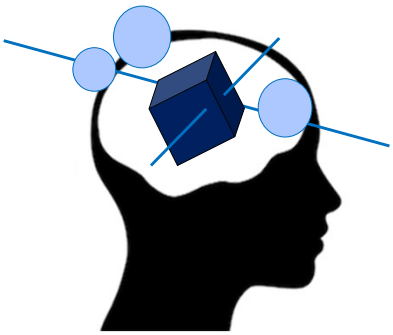
3D videogames

Mathematical representations of roto-translations

(a brief note)

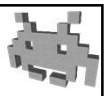




Marco Tarini

140

Representations for roto-rotations (just a brief note)

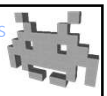


- So far, we assumed that the **rotation** and **translation** components of a transformation are stored *separatedly*
  - We have seen reasons why this is convenient
- Mathematical representations exist, that store rotation + translation (aka *roto-translations*, aka *rigid* transformations) jointly:
  - 4x4 matrices (we have seen them, their pros and cons)
  - Dual quaternions

141

Representations for ~~rotations~~ roto-translations

aka "rigid" transforms



- 3×3 Normal Matrices
- Euler Angles
- Angle & Axis
- Quaternions

+ Translation (displacement vector)

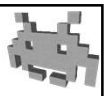
OR:

- 4×4 Matrices (or 3×4)
- Dual Quaternions

As there's no need to store the last row, it's (0,0,0,1)

142

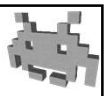
The math of Dual Quaternions in a nutshell 1/3



- Dual quaternions are a mathematical way to represent a roto-translation (aka, a "rigid motion")
- They result in very good interpolation between 2 (or more) roto-translations
- They are sometimes used in animation techniques
  - See lecture about skeletal animations, later

144

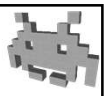
The math of Dual Quaternions in a nutshell 2/3



- New "fantasy" assumption:
  there is a ε such that
  $\varepsilon \neq 0$, $\varepsilon^2 = 0$
- A "dual number": $\mathbf{a} + \varepsilon\, \mathbf{b}$, with $\mathbf{a},\mathbf{b} \in \mathbb{R}$
- A "dual quaternion": $\mathbf{p} + \varepsilon\, \mathbf{q}$, with $\mathbf{p},\mathbf{q} \in \mathbb{H}$ (quaternion set)
- In can be stored as eight scalars
  (four for $\mathbf{p}$, plus four for $\mathbf{q}$)

145

The math of Dual Quaternions in a nutshell 3/3

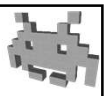


- A dual quaternion $(\mathbf{p} + \varepsilon\, \mathbf{q})$ can represent
  (under distinct conditions):
  - a 3D point, OR
  - a 3D vector/versor, OR,
  - a roto-translation
    then, the "primal" quaternion $\mathbf{p}$ is the rotation, and
    the "dual" quaternion $\mathbf{q}$ encodes the translation (in some ways)
  - (OR, nothing)

  We won't see the details,
  but in case you'll ever need it,
  you'll find them in the course web page
- We can apply a roto-translation to a 3D point or vector,
  by simply manipulating their representations
  - With multiplications & conjugations
  - Always keeping in mind all the "fantasy assumptions"
  - Same principle of representing 3D rotations with quaternions
    (or 2D rotations with complex numbers)

146

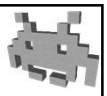
Q: why dual-quaternions?
A: better interpolation of rigid motions



- Problem with interpolating rotations and translations separately (intuitively):
  - must choose “which one goes first” (R then T, or, T then R)?
  - Different choices ⟶ very different interpolation results
  - Often, neither is what we had in mind
- Dual quaternions = a better* math abstraction to model roto-translations
  - “better” as in: it produces better interpolations

147

Dual Quaternions



- We won’t see the details of the math of dual quaternions.
- But in case you’ll ever need it, you’ll find all the ready-to-use details at the in the course web page

148