Practical Session #10

Project Part 3

Interaction & Animation

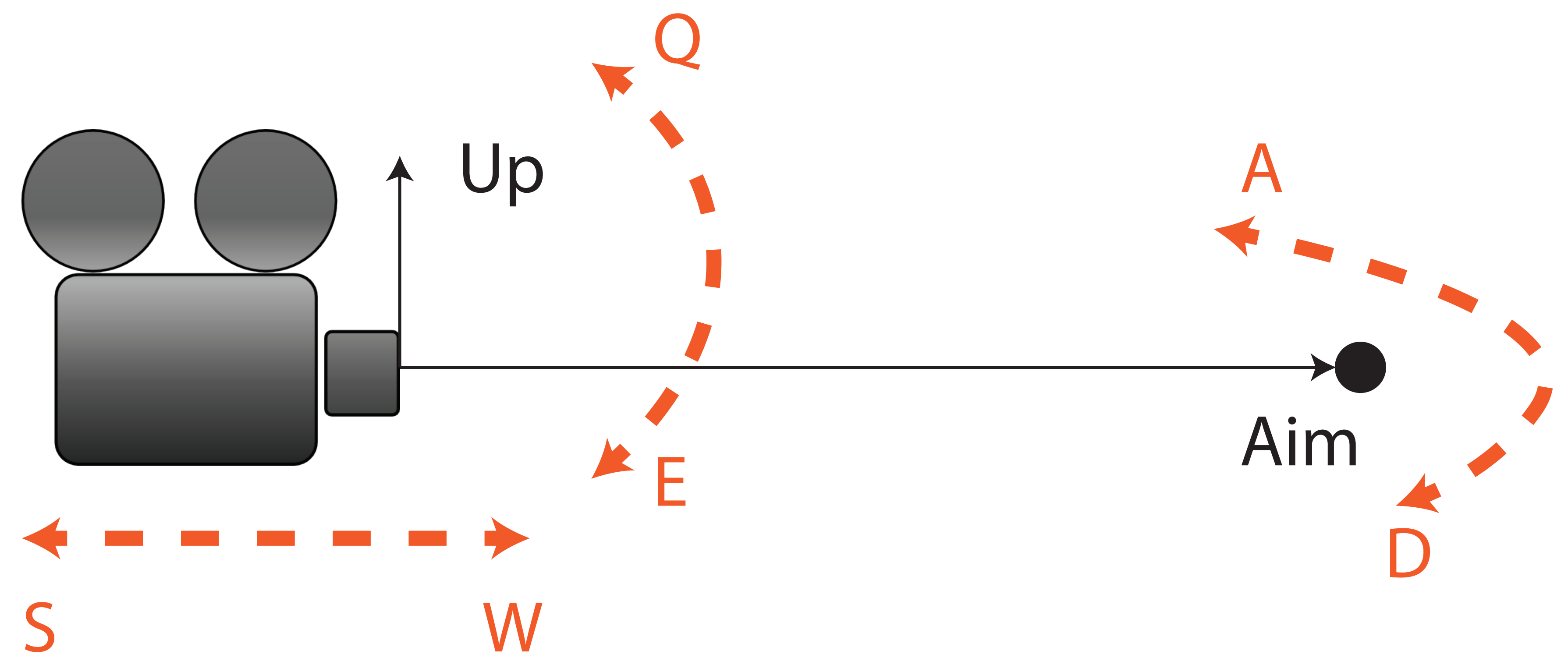
Basic topics (0..4/6)

3 points: implementation of technique
1 point: visual quality of results

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

1. Basic camera control

Perspective camera defined by:
- camera position cam_pos
- camera aim (or look-at point) cam_look
- up direction cam_up



To generate View matrix:
View = Eigen::lookAt(cam pos, cam look, cam up)

Task 1: Fly-through mode: Change position, aim and up direction by key strokes
- Use W and S to move the camera along the view direction
- A and D to rotate the aim about the up direction
- Q and E to roll the camera up and down
- Add inertia to your camera controls
  - e.g. after pressing W, the camera velocity increases by a certain amount then decreases over time to 0
- Example: http://lgg.epfl.ch/teaching/icg/fly_over.mp4
- Hint: See HW4 for an example of keystroke handling

1. Basic camera control

Task 2: First-person shooting (FPS) exploration mode:
Control the x and y components of the camera while snapping the camera to the corresponding height of the terrain.

Use the same keys as the previous section to control the camera

Hint: You will need to query the terrain height at a position (x, y).
Use glGetTexImage(…) or glReadPixels(…) to access the generated height map from FBO.

2. Camera path control

Task 3: Use Bezier curves to control the camera path

In HW5 (bezier): you have implemented the basics

- edit the control points of a bezier curve
- animate the camera by sampling two curves for the camera position and aim

Task 3.1 Design a nice camera path to explore your terrain.
You will be graded based on the visual quality
Example: http://lgg.epfl.ch/teaching/icg/camerapath.mp4

Task 3.2 Control the camera velocity along the path.
The camera velocity in HW5 is constant. Change the velocity by keystrokes: W to increase and S to reduce the velocity.

Advanced topics (4..6/6)

Idea 1: Physically realistic movements of the camera

- Objects don't accelerate instantaneously as they have mass
- Build a simple physical system to emulate a physically plausible motion (possibly integrating jumping)

Idea 2: Bezier curve

- Instead of using a fixed recursion depth for "de Casteljau", implement adaptive recursion depth
- For more details: http://algorithmist.net/docs/subdivision.pdf

Idea 3: Ease-in/out camera movement

- Use another bezier curve to control the camera velocity along the camera path.

Idea 4: Particle System

- Example: http://lgg.epfl.ch/teaching/icg/Snowing.mov
- Refer to the accompanied pdf for technical details.