Forward Kinematics

2020/5/3

Outline

- Project overview and demo
- Objective and explanation
- Score criteria
- Submission
- Hint and Reminder

Project Overview

- Understand representations of motion capture data
  - asf：acclaim skeleton file
  - amc：acclaim motion capture
- Implement forward kinematics of an hierarchical articulated system
- Time warping

Forward Kinematic

- Covert motion data from joint space to the Cartesian space

- Reference
  - acclaim_FK_IKnote.pdf, p.1-p.4
  - kinematics.pptx, p.6-p.19

Time Warping

- Modify the given motion sequences with arbitrary profile

- Reference
  - keyframing.pptx, p.40-p.50
  - acclaim_FK_IKnote.pdf, section “Motion interpolation”, on p.7

Environment

- Visual studio 2017 / 2019

- Third party
  - fltk：gui
  - Eigen：matrix operations
  - boost：useful utilities beyond STL

gui Module

- gui::MainWindow
  - Specify the UI window
  - Handle all UI flows

- gui::Display
  - Handle almost all events from MainWindow
  - Draw skeletons

gui::MainWindow

- LoadSkeletonSlot
  - slot function to the button Load skeleton
- LoadMotionSlot
  - slot function to the button Load motion
- TimeWarpSlot
  - slot function to the button Time warp

gui::Display

- Contain loaded skeletons & motions

- Access asf & amc data by integer index
  - skeleton_coll_->at(i) the i-th loaded asf file
  - motion_coll_->at(i) the i-th loaded amc file
  - fk_solver_coll_->at(i) the forward kinematics for the i-th skeleton & motion

math Module

- Contain all necessary vector and matrix operation
- Implemented with *Eigen*
  - A C++ template library for linear algebra
  - Eigen website：
    http://eigen.tuxfamily.org/index.php?title=Main_Page

math Module

- math::Vector3d_t
  - Column vector with x, y, and z components
- math::RotMat3d_t
  - 3D rotation matrix
- math::HomoXfm3d_t
  - 3D homogeneous transformation
- math::Quaternion_t
  - 3D quaternion with w, x, y, and z

math Module

- math:: Vector6d_t
  - Column vector containing 3D angular & linear vectors
  - Represent the 6 degrees-of-freedom in the Cartesian space
  - Only root joint has translations

- math:: Vector6dColl_t
  - Represent motion data of a single frame

math Module

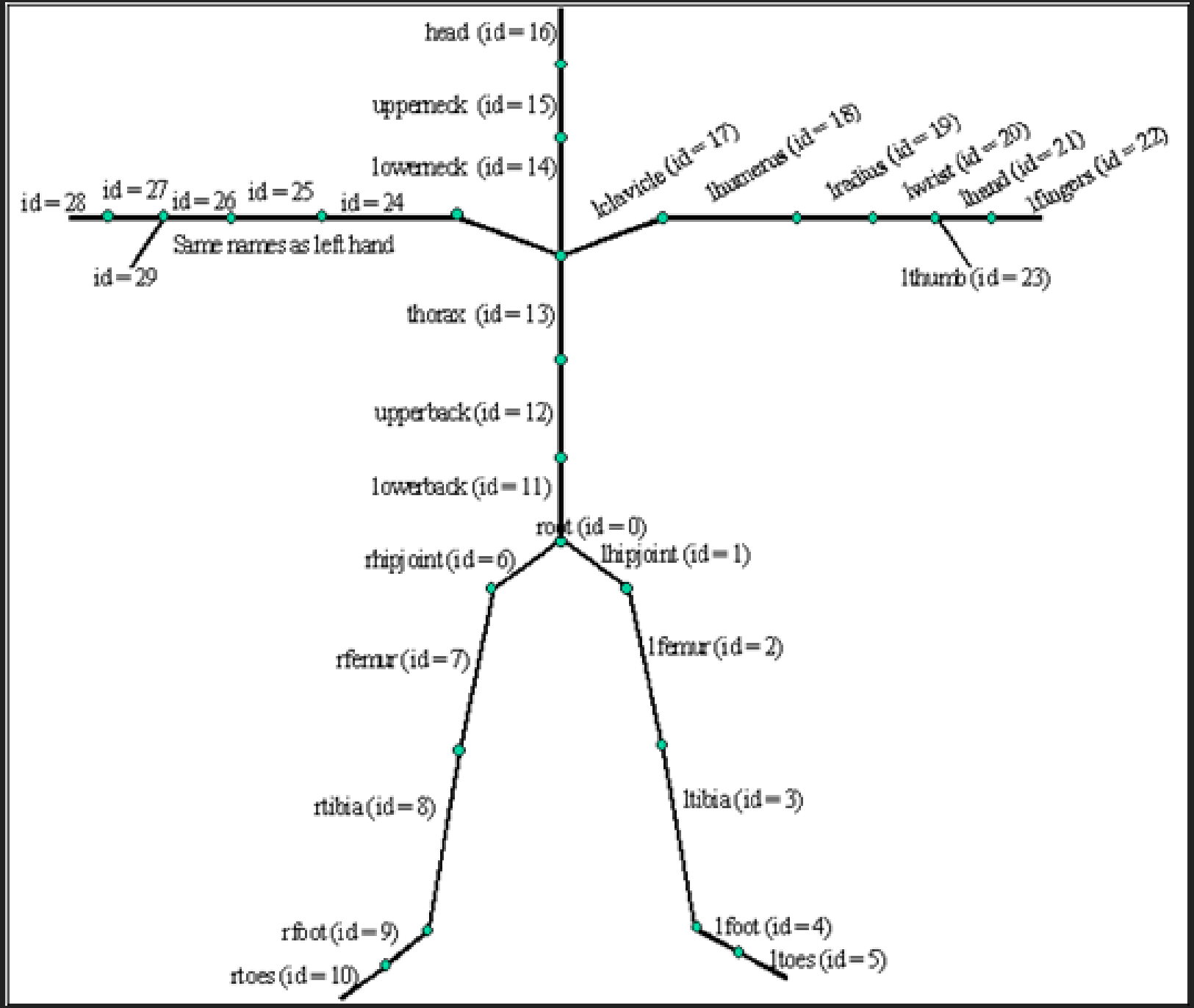
- math:: SpatialTemporalVector6d_t
  - 2D storage of math::Vector6d_t
  - Represent whole motion sequence of an amc file
  - Temporal_size is the frame no. of the amc file
  - Spatial_size is the body no. of the asf file

math utilities

- math::ComputeRotMatXyz
  - Compute rotation matrix from Euler angles by the order x-y-z
- math::ComputeQuaternionXyz
  - Compute quaternion from Euler angles by the order x-y-z
- math::ComputeEulerAngleXyz
  - Compute Euler angles from a rotation matrix by the order x-y-z

kinematics Module

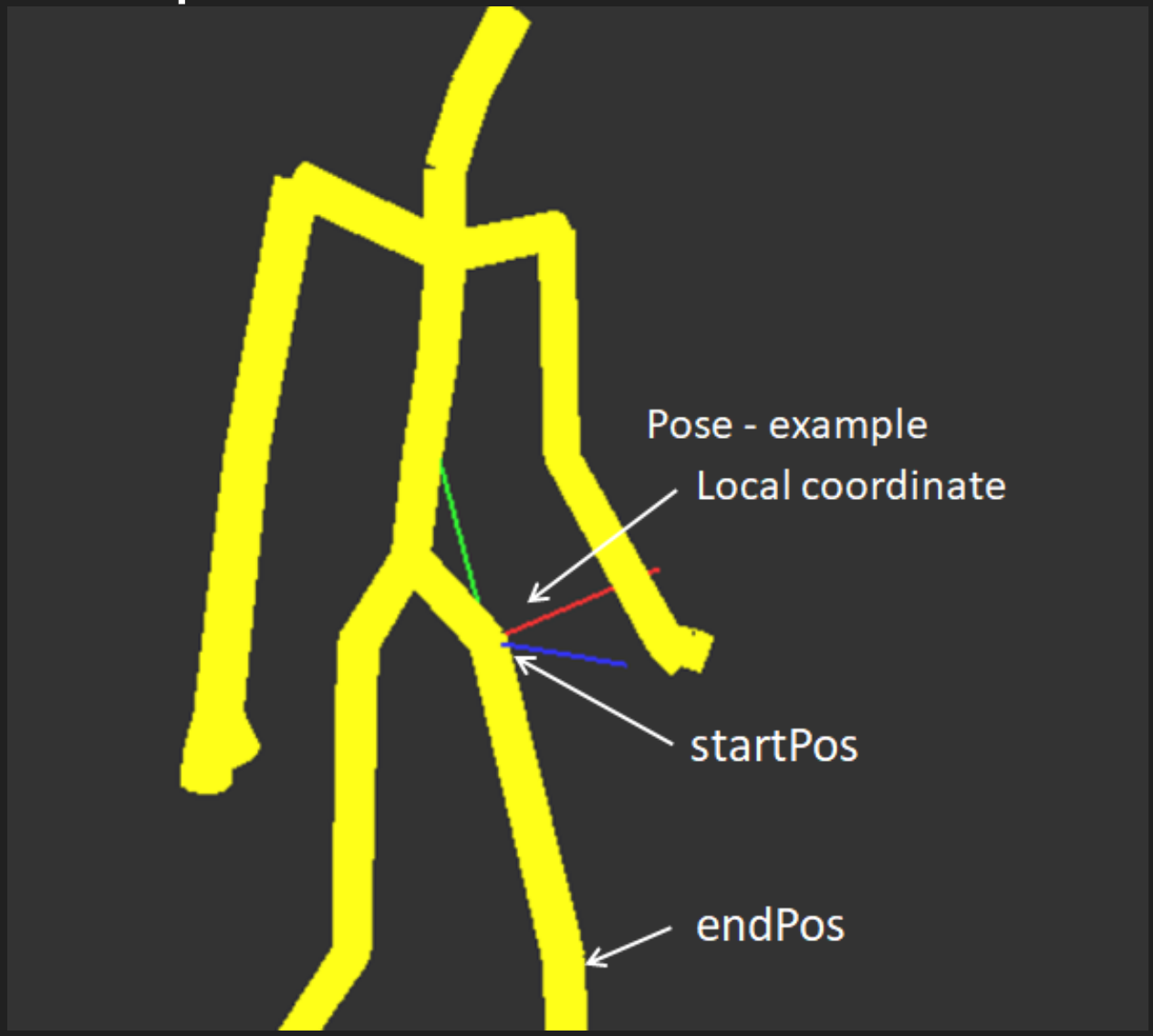
- kinematics::ArticIdx
  - Represent the body index with its parent body index
  - Used to construct paths to traverse the articulated system
    e.g., root -> lhipjoint -> lfemur -> ltibia -> lfoot


head (id = 16)
upperneck (id = 15)
lowerneck (id = 14)
id = 28
id = 27
id = 26
id = 25
id = 24
Same names as left hand
id = 29
lclavicle (id = 17)
lhumerus (id = 18)
lradius (id = 19)
lwrist (id = 20)
lhand (id = 21)
lfingers (id = 22)
lthumb (id = 23)
thorax (id = 13)
upperback (id = 12)
lowerback (id = 11)
root (id = 0)
rhipjoint (id = 6)
lhipjoint (id = 1)
rfemur (id = 7)
lfemur (id = 2)
rtibia (id = 8)
ltibia (id = 3)
rfoot (id = 9)
lfoot (id = 4)
rtoes (id = 10)
ltoes (id = 5)

kinematics::Pose

- Represent the bone’s translation & rotation
  - start_pos：the proximal end
  - end_pos：the distal end
  - rotation：the rotation matrix
- Show forward kinematics result by whole-body pose
  - apply gui::Renderer::DrawCylinder

Pose example



acclaim Module

- acclaim::Skeleton
  - data structure for the parsed asf file

- acclaim::Motion
  - data structure for the parsed amc file

acclaim::Skeleton

- bone_ptr
  - Access the acclaim::Bone instance
  - Refer to “acclaim_FK_IKnote.pdf” for detailed definition of each variable

- bone_num
  - Bone no. of the skeleton instance

acclaim::Motion

- joint_spatial_pos
  - Access the motion data of the indicated frame

- whole_sequence
  - Access all motion data contained in the amc file

param Module

- param::Config
  - Parse the parameter setting in the .\parameter\config.xml

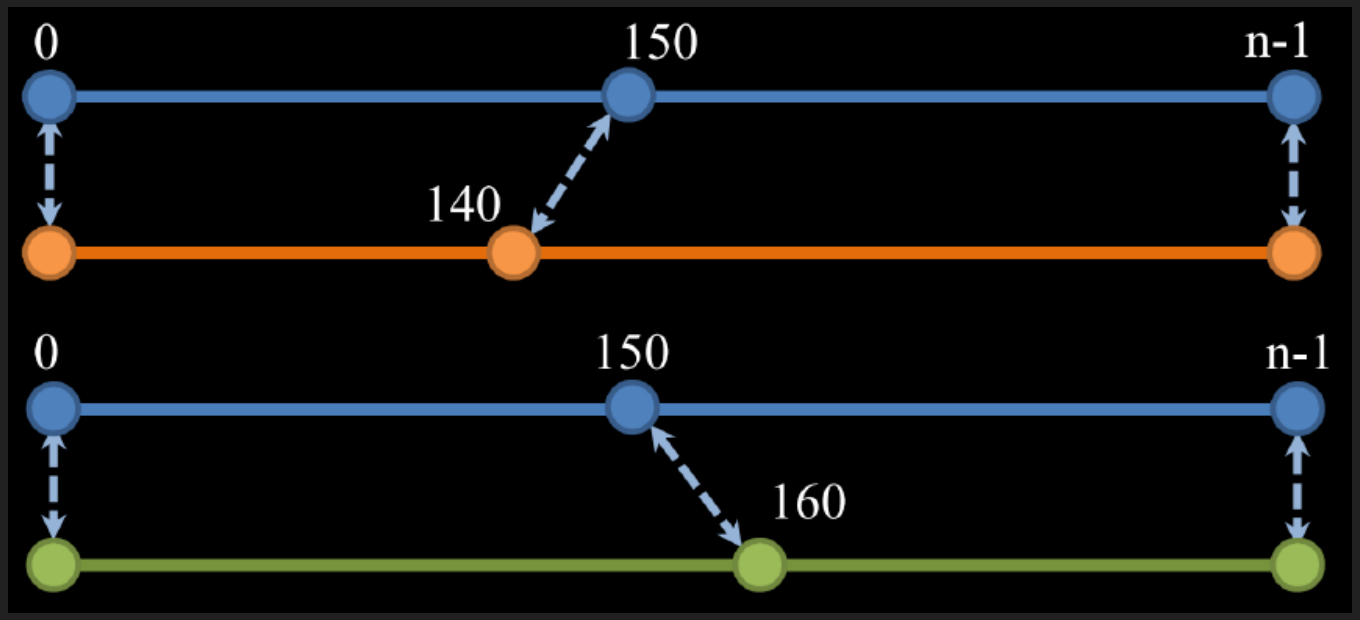
Forward Kinematics - What to Do

- Load “running.amc” or “punch_kick.amc”
- Implement kinematics::ForwardSolver
  - Remove the instance helper::ForwardKinematics
- ComputeSkeletonPose
  - Input：motion data of a single frame (math::Vector6dColl_t)
  - Output：whole body pose (kinematics::PoseColl_t)

Time Warping

- kinematics::TimeWarpHardConstraint_t
  - frame_idx
  - play_second
- Two sets of constraints
  - h: time step, n: frame no.
  - {0, 0.0}, {140, 150h},{n-1, (n-1)h}
  - {0, 0.0}, {160, 150h}, {n-1, (n-1)h}
  - Access the constrained frame_idx via param_->value<int32_t>("time_warp.desired_catch_frame_idx“)

- h：time step, n：frame no.
  - {0, 0.0}, {140, 150h},{n-1, (n-1)h}
  - {0, 0.0}, {160, 150h}, {n-1, (n-1)h}



Time Warping - What to Do

- Load “punch_kick.amc”
- Implement kinematics::TimeWarper
- ComputeWarpedMotion
  - Input：hard constraints
  - Output：whole sequence of warped motion
- Your character should catch the ball after warping
  - Pass two sets of constraints listed on the previous page

Interpolation

- Use Quaternion
- Linear interpolation

Bonus

- Do motion transform in Time Warping part
- Rotate the root for 180 degrees, then apply the translation of (ball pos - root) * 2 to place it at the opposite side of the ball
- Translation Just consider the horizontal difference of the ball and root
- Reference
  - acclaim_FK_IKnote.pdf, p.5-p.6

Score criteria

- Froward kinematics：60%
- Time warping：30%
  - Using Quaternion
- Report：10%
- Bonus : 20%

Suggested Outline of Report

- Introduction/Motivation
- Fundamentals
  - Describe local & global coordinate in your words
- Implementation
- Result & Discussion
- Conclusion

Submission

- Compress all materials into a zip file
  - Ensure your solution could be built successfully
  - Naming rule: CA2_ID_Version
    e.g., CA2_1234567_v01.zip
- Your zip file shall contain
  - Source code (without boost & Eigen)
  - Report in pdf or MS word format, no more than 10 pages
- Update to E3
  - No limit to the no. of times you can upload
  - The latest version is your final submission

Due Date

- 2020/5/18, 23:55

Hints and Reminder

- DO NOT include any Chinese/Mandarin characters in the path to your project
  - The program will crash when loading files
- A lib of compiled FK solver is provided
  - Remove helper_fk_ and related code in ForwardSolver
  - If you cannot complete the FK part, you can try to implement time warping with the helper left in (this way you won't get the score for FK)

Hints and Reminder (cont'd)

- Please run the source code under release mode
- How to contact TAs?
  - Please send email via new E3
  - IMPORTANT：please sort out and arrange your questions, so we can help you without wasting time on trivial matters