Comparing Motion Editing Methods

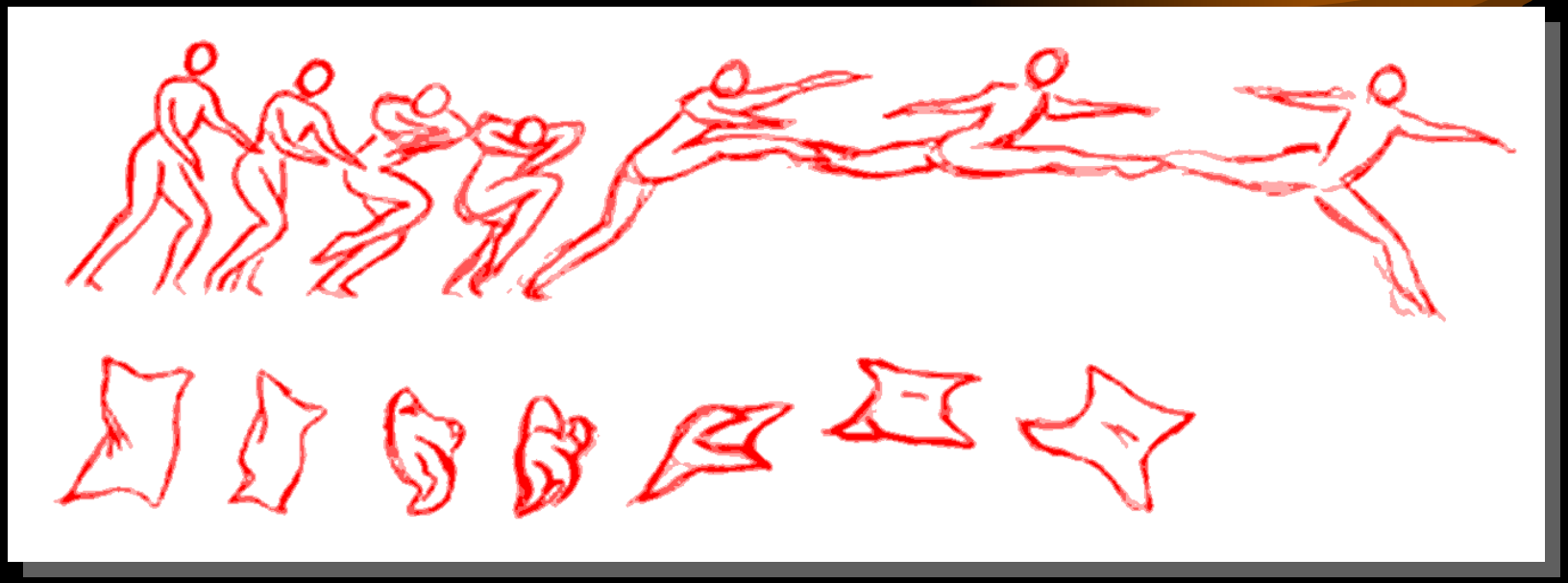
Michael Gleicher
Department of Computer Sciences
University of Wisconsin- Madison

Summary

- There's a wide variety of things to do…
- All have their downsides…
- The tradeoffs might not be what I thought…

- Spacetime is expensive (but how?)
- You get something for your efforts (what?)
- Other options (which?)

Motion Editing?



- Changing the Movement (not the form)
- Pretty unique for computer animation.
- Could be just about anything…

Motion Editing

- What changes? What doesn't change?

- Methods characterized by the range of how they can answer these questions
  - Implicitly: methods just can't make some changes (e.g. timing only)
  - Explicitly: allow specifying what's important: Constraints

A Taxonomy of Motion Editing

- Taxonomy by what can be done
- Taxonomy by how it does it
- Taxonomy by what it can do it to

Why?

Help understand tradeoffs

Point towards unexplored areas of design space

Constraint-Based Motion Editing

- Focus: methods that *explicitly* represent *geometric* constraints
- Independent of parameters
  - E.g. specify end-effector goals too
- Such methods must:
  - Deal with non-linear mappings
  - Deal with temporal constraints
  - Address issues in specification, display, …

Taxonomy of Solution Methods

- What is “under the hood”?
- Who cares?
  - Me (since it’s what I do)
  - Other issues independent
  - Should the user care? (if not, should we care?)
  - Scariest part of implementation (?!?!?!?)
  - Other challenges really given by systems/integration issues

Types of Constraints

Spatial Constraints

- Geometric Details
- Checkable at times
- Possibly over durations
- Limitations of character, interactions with environment
- Joint limits, footplants, hand-holds, intersection, …

Temporal Constraints

- Restrictions on how things move
- Always relate times
- About paths
- Smoothness, Continuity, …

Taxonomy by:
How are Temporal constraints handled

- By definition, constraint-based techniques handle spatial constraints
- Well studied problem

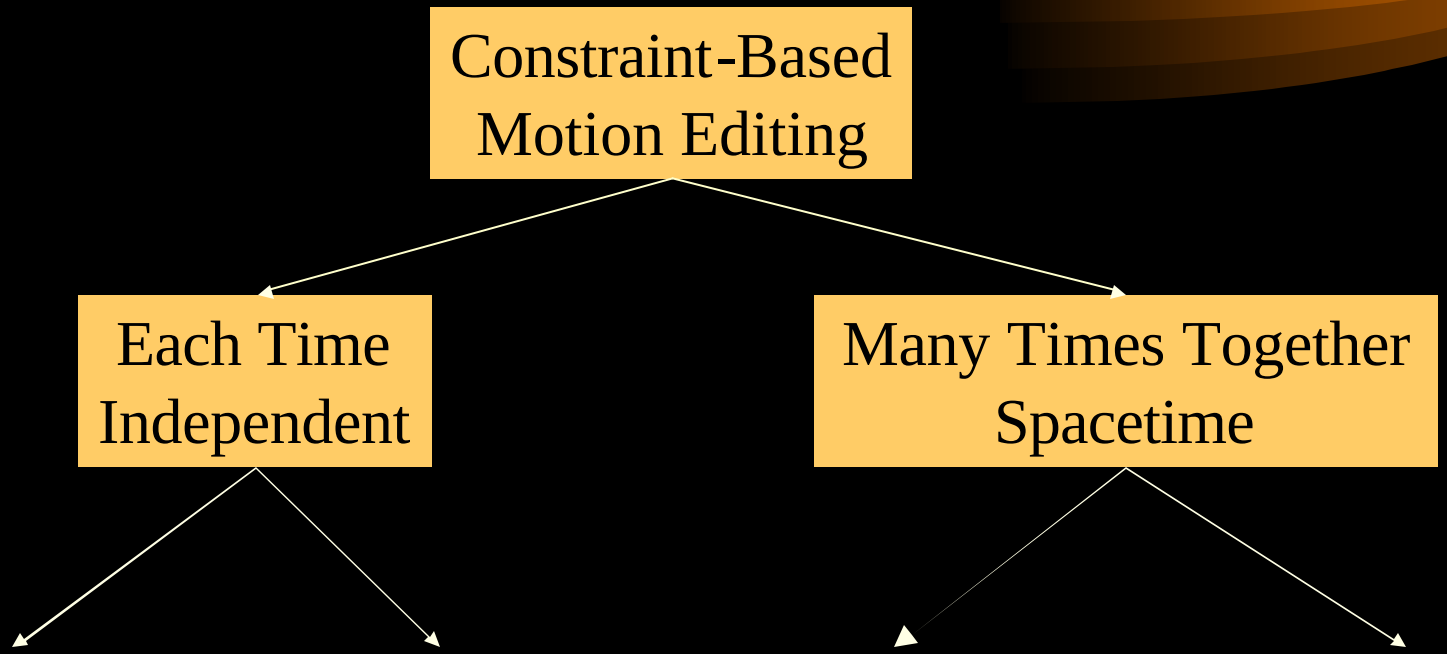
- Temporal constraints differentiate approaches

What Temporal Constraints?

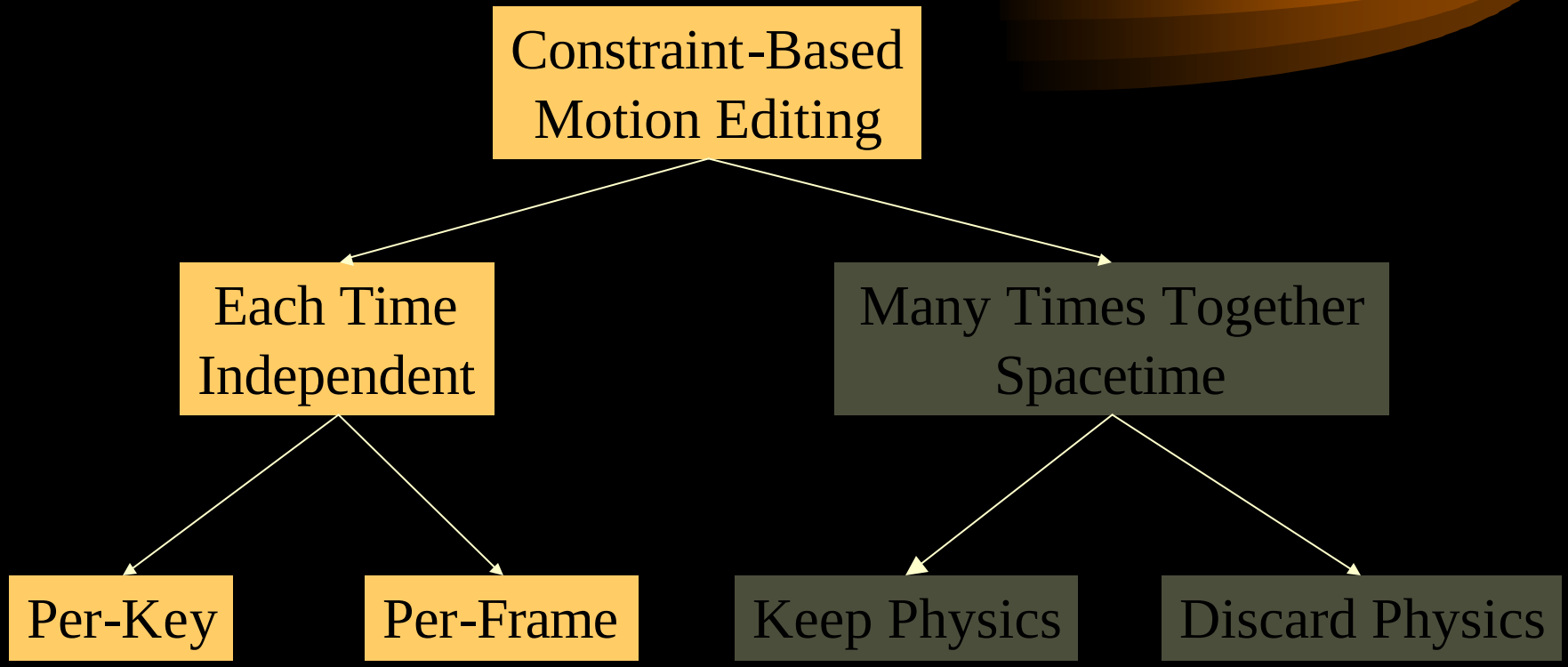
- Not as well defined as spatial ones!

- Preserve frequency content
  - Don’t add jitter / jumps
  - Don’t remove snappiness
  - Keep solutions consistent
- Preserve physics
- ????? (what else?)

Taxonomy of CBME solvers



Taxonomy


Constraint-Based Motion Editing
Each Time Independent
Many Times Together Spacetime
Per-Key
Per-Frame
Keep Physics
Discard Physics

Per-Frame vs. Per-Key

- Apply an “IK” solver to compute parameters for individual time steps

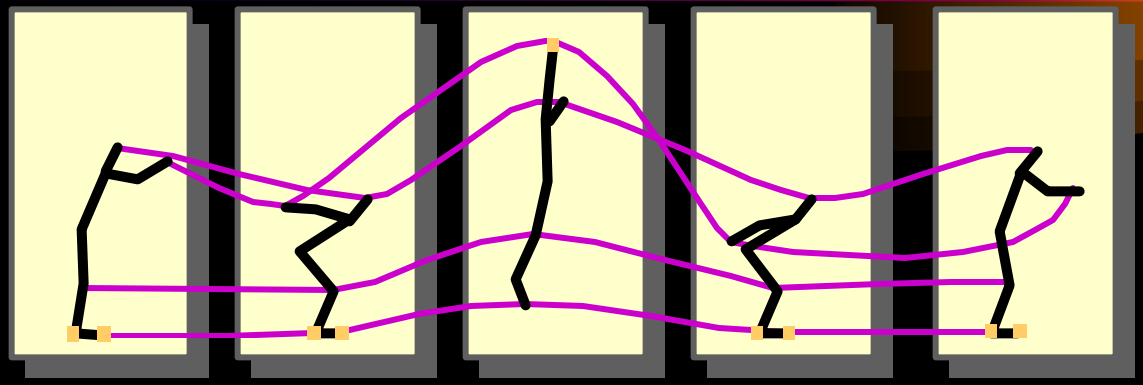
- On each key (e.g. sparse) PKIK
- On each frame (e.g. dense) PFIK

- Solver computes 1 instant
  - May consider other times in doing so

Per-Key vs. Per-Frame

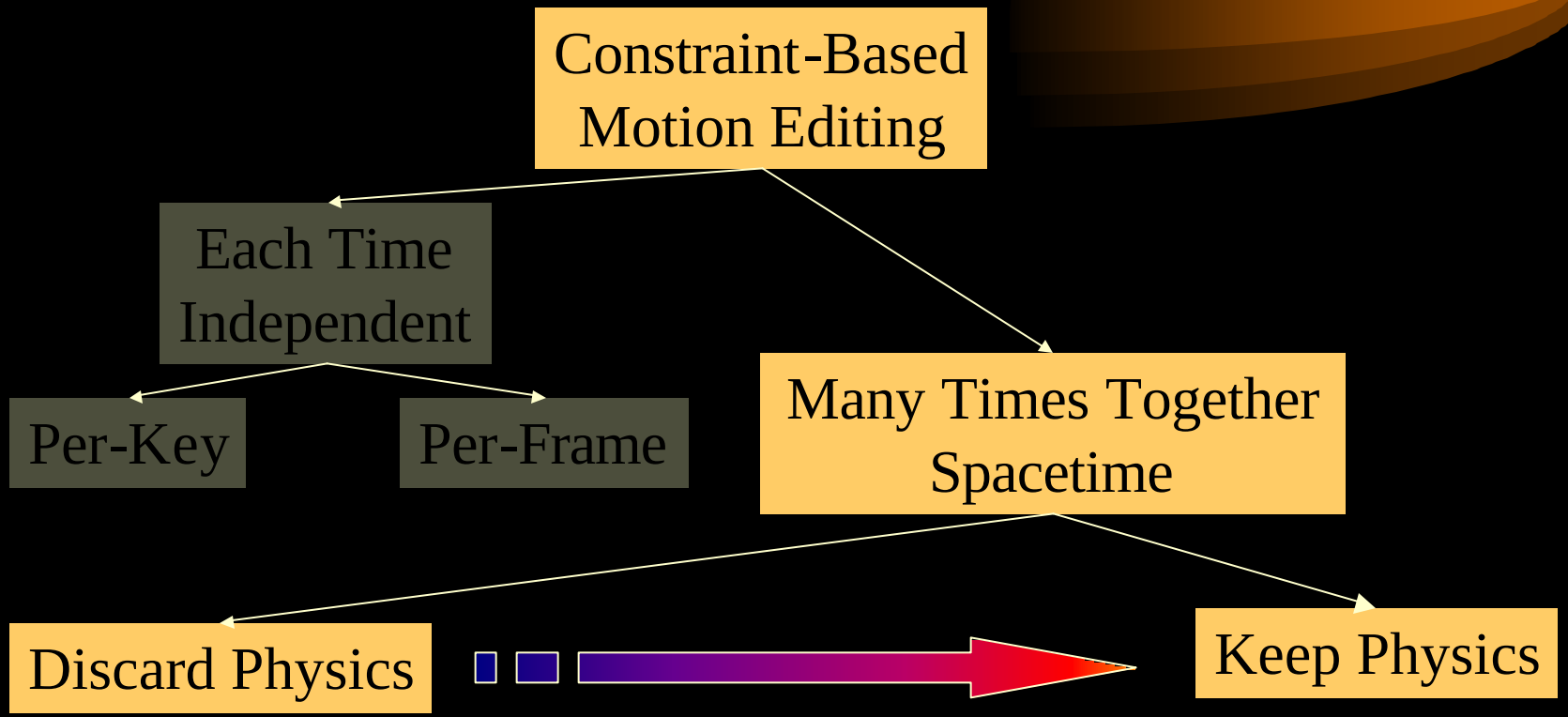
- Per-Key implies interpolation
  - Continuity through interpolation
- Per-Frame requires some way to keep consistency (avoid inducing jitter, …)
  - Look-behind
  - Compute one frame, from many
  - Build smarts into IK solver

Global (spacetime)



- Consider all constraints simultaneously
  - NOT frame at a time
- Solve for motions
  - “best” motion that meets constraints
- Physics is just a constraint

Taxonomy



Simplify for tractability

Complex constraints for Quality

Each for something different

- PKIK - requires (meaningful) keys
- PFIK - places burden of temporal constraints on IK solver
- Spacetime – off-line, complicated, hard to implement, but gives nice results?
  - Physics spacetime? Really hard to do, slow, unscalable, not demonstrated, ….

Video: Wins and Loses of Spacetime

- "Found" examples (retarget found motion to found character)
- Path Transformations (unpublished)
- Real-time, interactive examples

- Bloopers
  - (note: Pelican is first try- we can probably get it to work)

Wins and Losses of Spacetime

- Fast, practical
  - Linear complexity?
- Solves real problems
- Flexibility in:
  - Spatial Constraints
  - Objective Functions ?
  - Temporal Constraints ?
- Widely applicable
- Nice results

- Hard to implement
- Poor integration
- Off-line
- No guarantees
- Spatial constraints not enforced
- Flexibility not exploited
- Rely on constraints

Is there an alternative?

- Need to deal with spatial and temporal constraints
- Don’t want the messiness of “whole motion” computation

- Handle spatial and temporal constraints separately!

Per-Frame IK + Filter (PFIK+F)

- IK per frame to solve spatial constraints
  - But this messes up temporal constraints
- Filter changes to enforce temporal constraints
  - But this messes up spatial constraints
- Iterate until converges, or …

PFIK+F in Practice

- The published instance of this approach is Lee and Shin (SIGGRAPH ’99)
- Many decisions to be made…
  - What IK solver? (tradeoff quality/performance)
  - How to do temporal constraints?
  - Representation of motion?
  - Iteration Schedule?
- L&S made innovative choices for each

PFIK+F vs. Spacetime

- Fast, practical
- Solves real problems
- Flexibility in:
  - Spatial Constraints
  - Objective Functions (?)
  - Temporal Constraints (?)
- Widely applicable
- Nice results

- Yes! (requires fast IK)
- Yes!
-
  - (depends on IK)
  - (depends on IK)
  - (limited, unexplored)
- Yes!
- Um, it’s a matter of taste, and IK quality

PFIK+F vs. Spacetime

- Use standard pieces!
- Use standard pieces!

- Choice in which last
- Solve spatial constraints last

- Need good IK

- Hard to implement
- Poor integration
- Off-line
- No guarantees
- Spatial constraints not enforced
- Flexibility not exploited
- Rely on constraints

My PFIK+F solver

- Use pieces I have lying around
- Non-linear optimizing solver for IK
- FIR linear filters for temporal constraints

- Not fast (numbers in paper are wrong)
- Not tuned
- (but L&S showed you can do this)

Video (Showdown)

- Unfair competition?
  - Well evolved Spacetime implementation
    - Although, it is used in PFIK+F as well
- Point: both methods can give similar results (differences are subtle)

Downsides of PFIK+F?

- No global decisions
  - Doesn’t handle “don’t cares” as well
  - Order dependence
  - No interframe constraints
- Reliance on quality of IK solver
- Not necessarily faster (or slower)

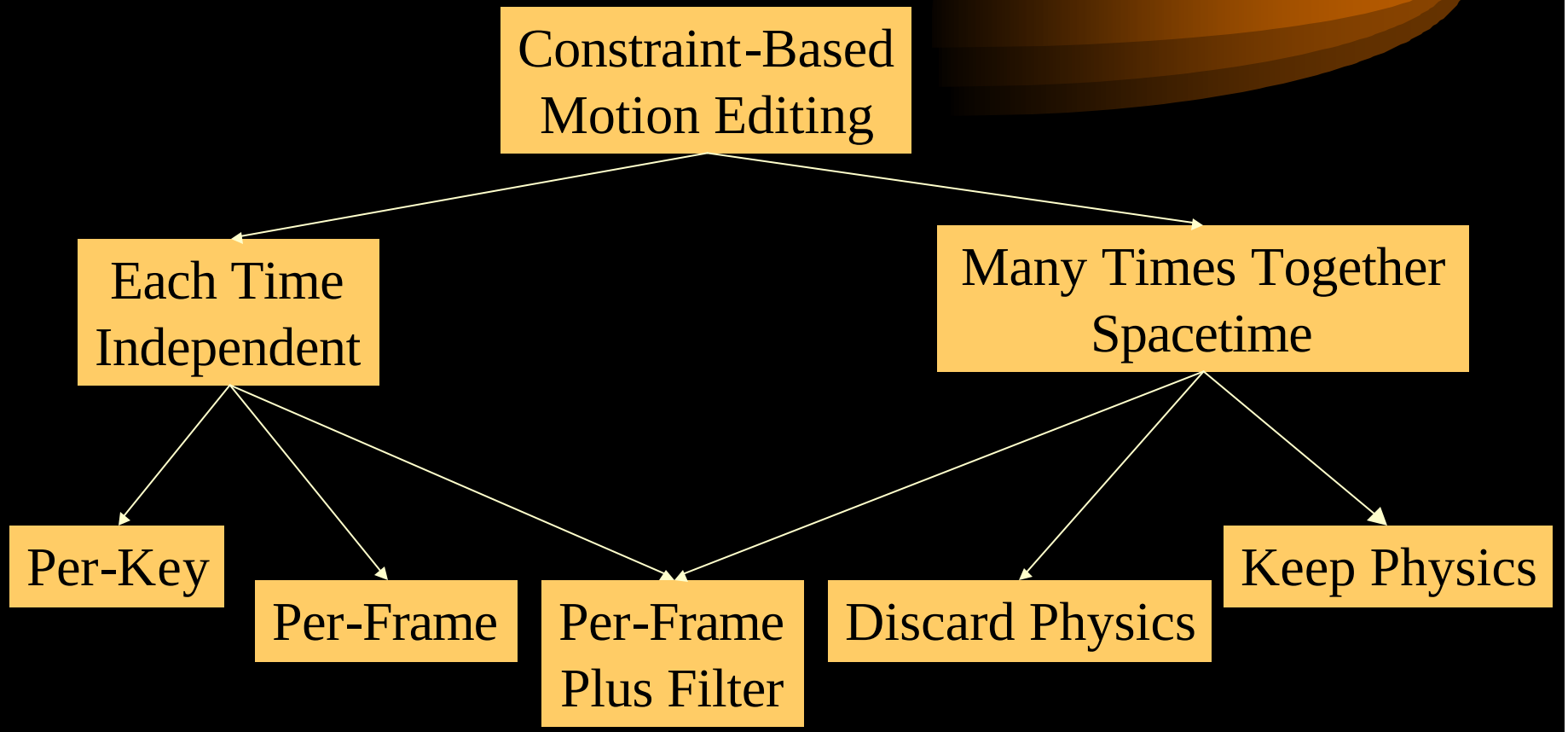
History of graphics performance tuning (from my short career)

- 1988 – (68020) avoid computation (cache)
- 1990 – (R2000) avoid floating point
- 1993 – (R3000) avoid array indexing
- 1995 – (PPC601) avoid type conversion
- 1997 – (PPC604) avoid memory allocation
- 1999 – (Pentium III) avoid cache misses

  (floating point is fast)
- 2000 – (Pentium III, Rambus) avoid memory stalls

  (memory is pipelined)

Constraint-Based Motion Editing



Summary

- There's a wide variety of things to do…
- All have their downsides…
- The tradeoffs might not be what I thought…

- Spacetime is expensive (but how?)
- You get something for your efforts (what?)
- Other options (which?)

Thanks!

- Microsoft Research for supporting this work.
- Intel, Pixar, Autodesk, and Alias/Wavefront for donating hardware and software
- The U.W. Visual Computing Group
  - Andy Gardner (for running trials, new demos)
  - Alex Mohr (debugging code and prose)
  - Andrew Prock (arc-length re-parameterizations)
- House of Moves Studios for providing data
- Prof. Ko for inviting me.
- Lori for letting me go