PSET 4 Review Part I

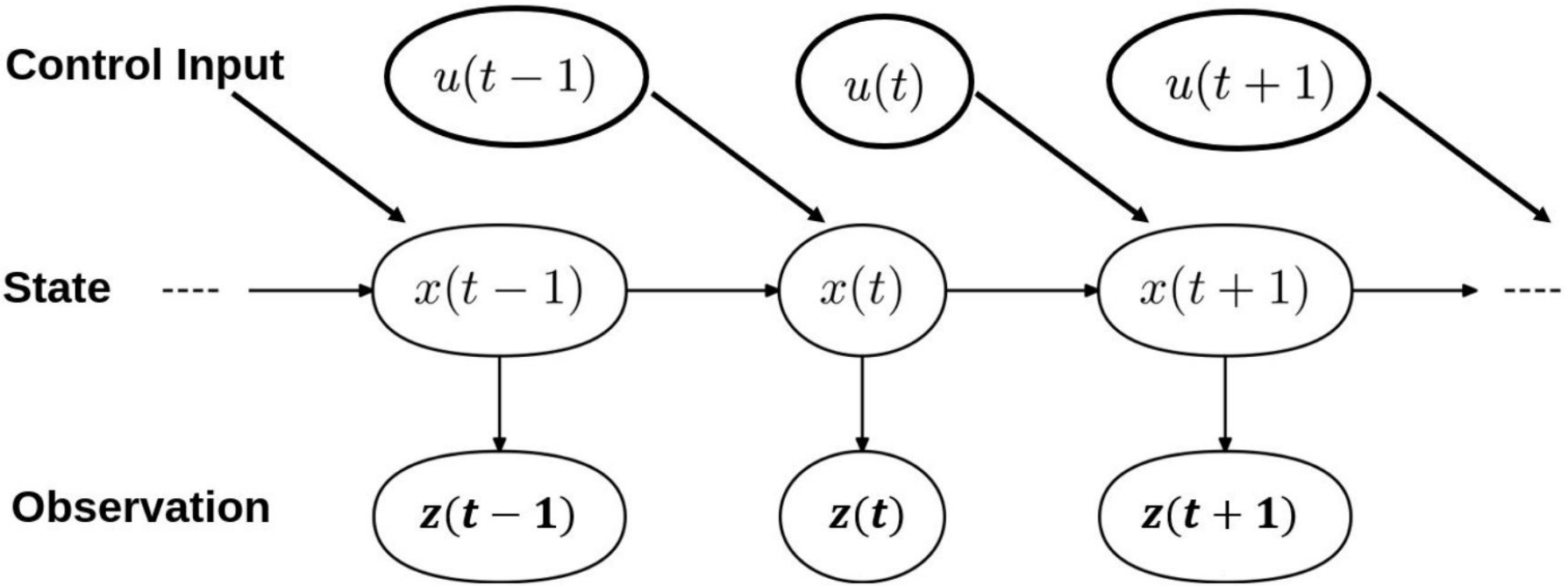
Krishnan Srinivasan

05/31/2024

Outline

- Kalman Filter (Getting started with problems 2-4)

Dynamical System

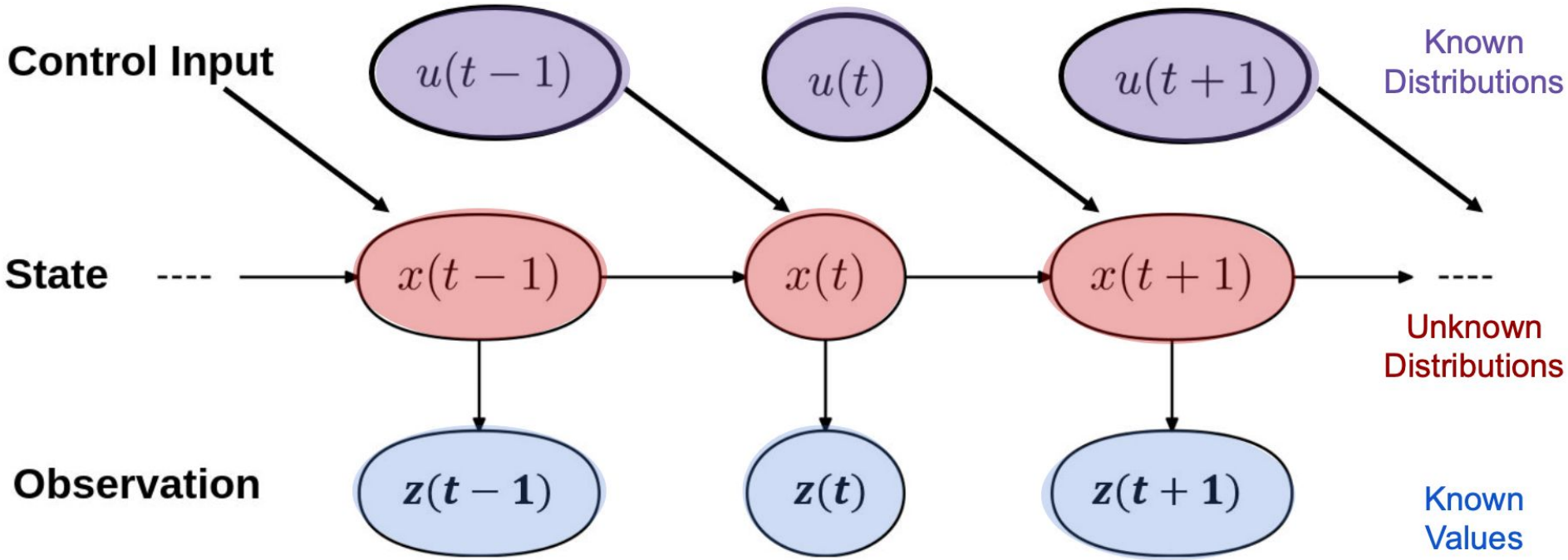


Kalman Filter

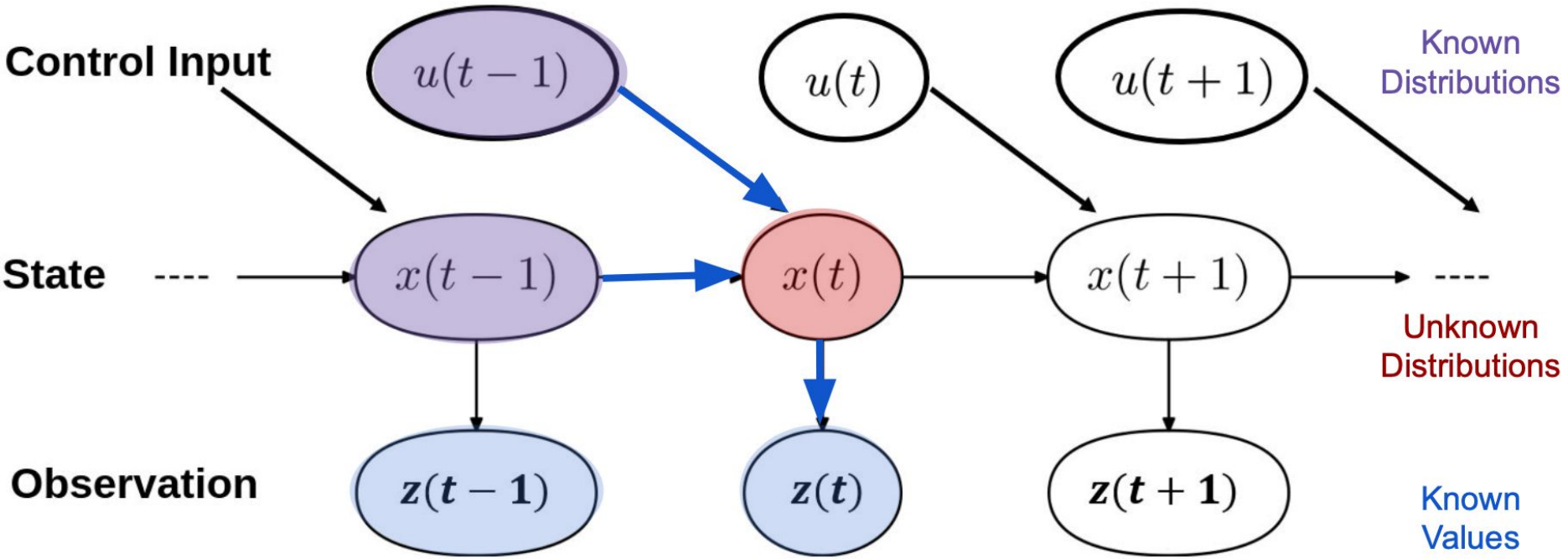
An algorithm that uses a series of **measurements observed over time**, containing **statistical noise** and other inaccuracies, and produces **estimates of unknown variables** that tend to be more accurate than those based on a single measurement alone, by estimating a joint probability distribution over the variables for each timeframe.

Source: Wikipedia

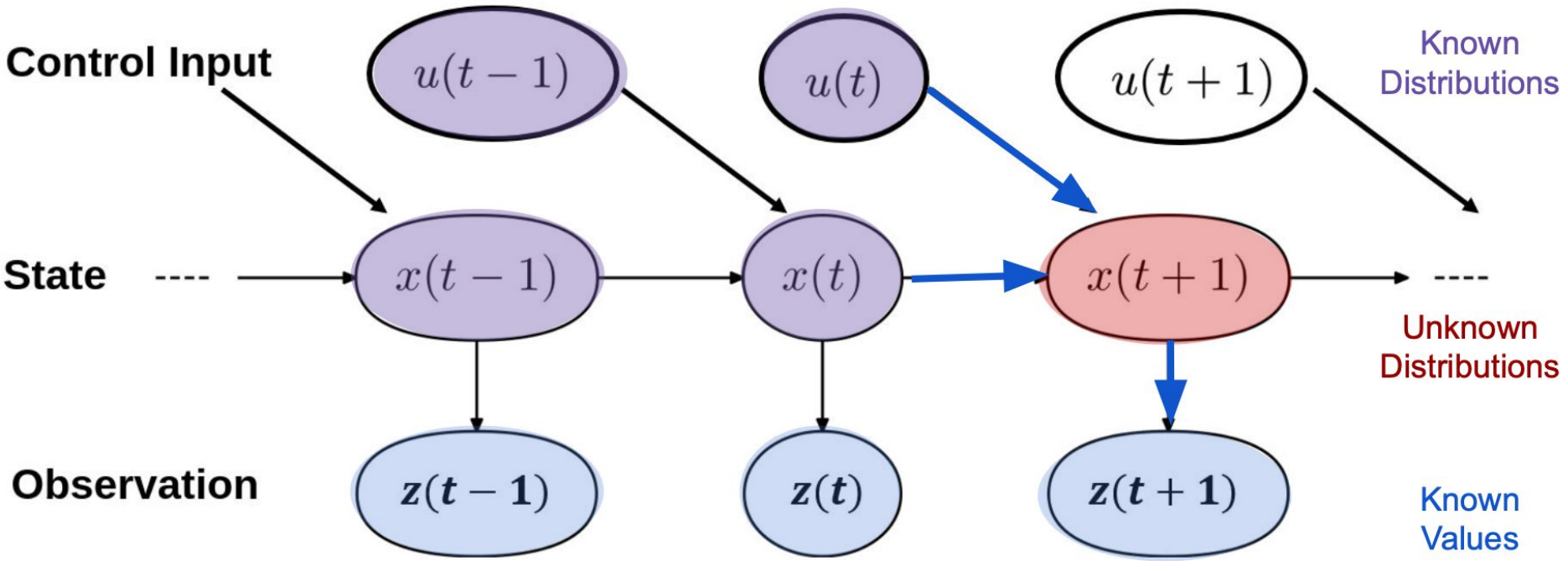
What does Kalman Filter do?



What does Kalman Filter do?



What does Kalman Filter do?



Problems with Kalman Filter

- Kalman Filters are defined for linear systems… but real life systems are complicated. What if they are nonlinear?
- Kalman Filters assume Gaussian distributions, which is preserved after linear transitions, but may not be preserved after nonlinear transitions, potentially causing KF to not converge.

Extended Kalman Filter

- Extended Kalman Filter (EKF) is heuristic for **nonlinear** filtering problem (nonlinear dynamics function, nonlinear observation model, etc.)
- Often works well (when tuned properly), but sometimes not.
- Widely used in practice.

Based on

- **Linearizing** dynamics and output functions **around the current estimate**
- Propagating an approximation of the conditional expectation and covariance through the linearized model

Implementing Extended Kalman Filter

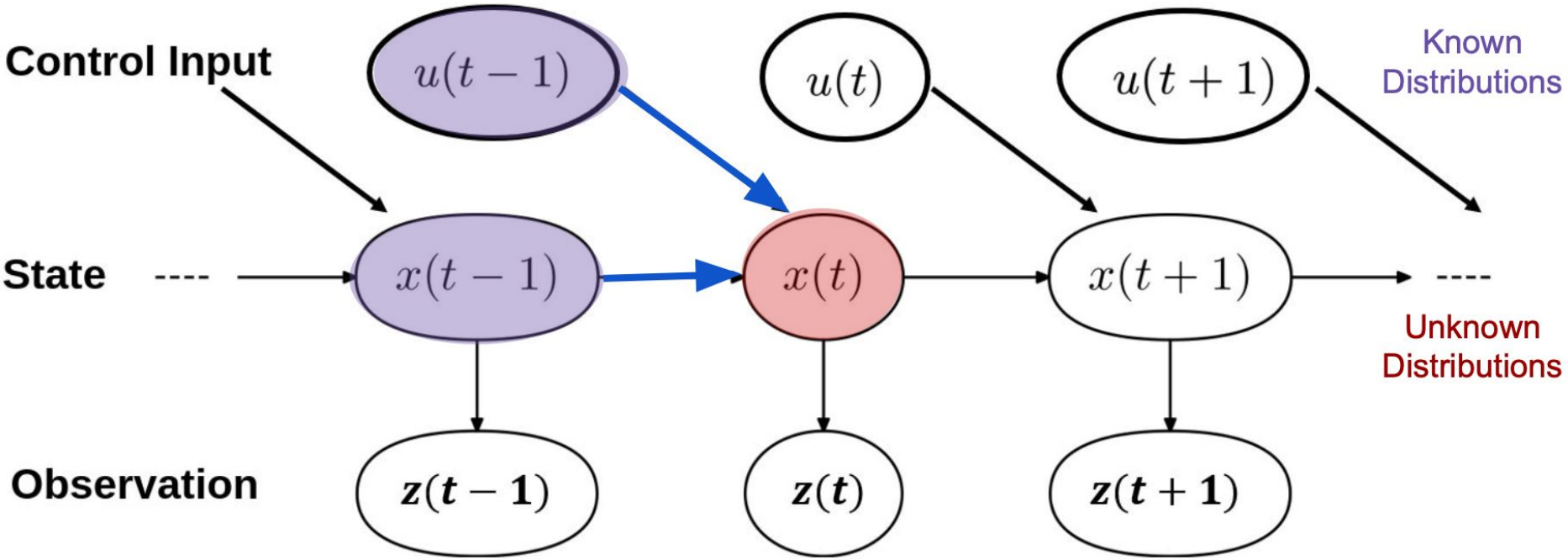
- Define the state, the control, and the noise
- Derive the system and the observation
- Compute the current Jacobian matrix (linearizing dynamics)
- Compute the distribution of the current state
- Iterate this process across time

Define the State

State: 6-dimensional vector (position, velocity)

$$x_t = \begin{bmatrix} p_t^x \\ p_t^y \\ p_t^z \\ v_t^x \\ v_t^y \\ v_t^z \end{bmatrix}$$

Define the System Matrix

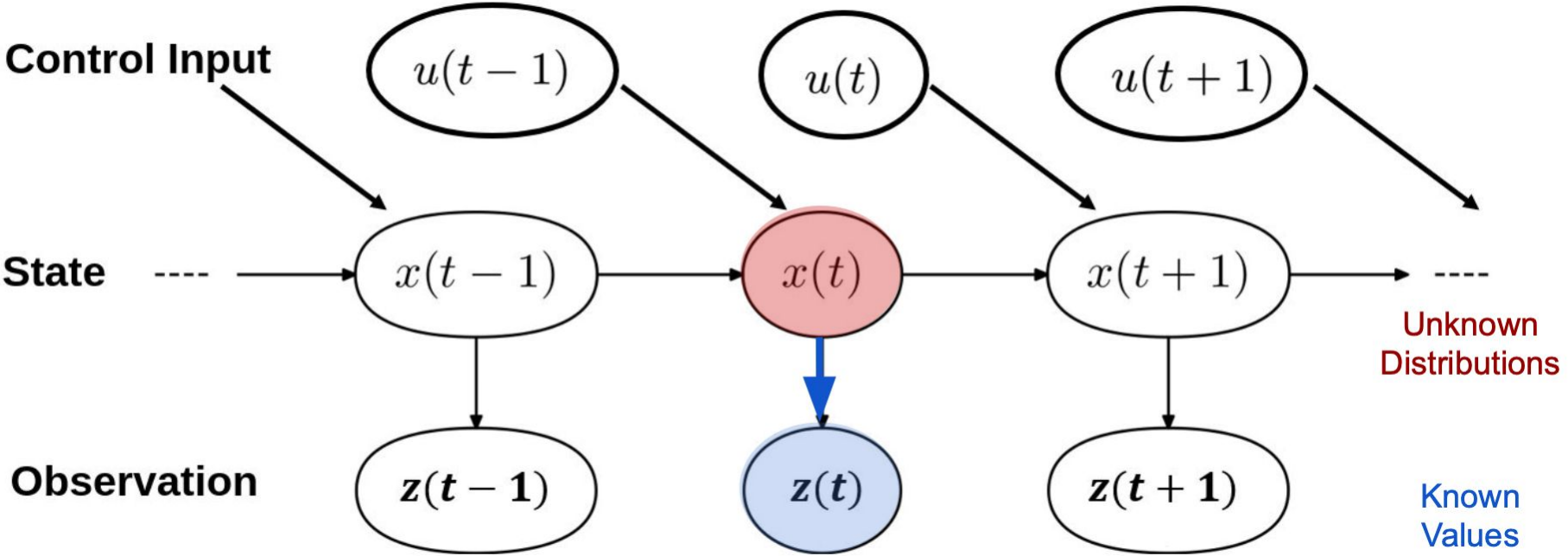


Define the System Matrix

$$x_t = \begin{bmatrix} p_t^x \\ p_t^y \\ p_t^z \\ v_t^x \\ v_t^y \\ v_t^z \end{bmatrix}$$

$$x_{t+1} = Ax_t + \epsilon_t$$

Define the Observation



Define the Observation

- Observation in Q2: pixel location (u, v)
- Observation in Q3: pixel location and disparity (u, v, d)
- What should h() be? Think about how we could derive it using the camera model and what we have learned in previous lectures.

$$\mathbf{z}_t = h(\mathbf{x}_t) + \mathbf{v}_t$$

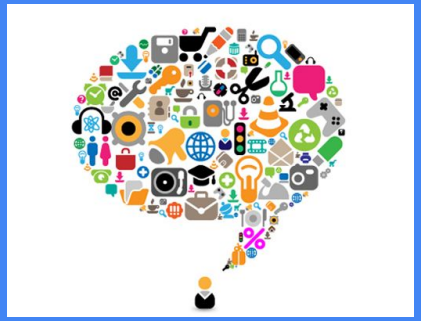
Final Project Presentation Guidelines

Krishnan Srinivasan
05/31/2024

Presentation Logistics

- Presentations will be no longer than 8 minutes with 1 minute for questions.
- Presentations will be held in HAI (First floor of Gates), from 1pm to 4pm
  - There is a survey asking what times you are available
  - If there are enough open slots we can move all projects to present within the same time window
- If making a poster:
  - Use this template
- If making a video presentation/attending virtually:
  - Make a Google slides presentation, shared with the CAs before presenting
- SCPD students can send a link to a 5 minute video instead (or by request)

Presentation Content



- Problem definition and motivation (1 min)
- Quickly: previous work
- Your solution with some technical detail (1.5 min)
  - What datasets do you use? How did you wrangle them?
  - What parts of the class did you use?
- Your experimental setup and preliminary results. (1.5 min)
  - We understand there's still a few days until the report is due
- Conclusions
  - What did you learn?

Presentation tips

- It's easy to put **too much** content into your talk.
  - We will cut you off at 4 minutes.
  - Have only enough content that you can talk slowly and still get through it
  - Make an effort to go slowly.
  - The best presentations only communicate the big ideas
- Communicate
  - The goal of your presentation is to communicate your ideas to your audience
  - The most impressive talks are the ones you understand, not the ones you don't
- No math
  - Math is almost impossible to get out of a talk, especially a short one
- Pictures really are worth a thousand words
  - When possible, show, don't tell

Presentation tips

- Practice
  - Practice helps. Practice now will even help your presentation skills in the future.
- I like to write down exactly what I plan to say in the speaker notes
  - Then I ignore them while giving the talk
  - Try not to read during your talk
- Turn nervous energy into enthusiasm!
- Don't be nervous about questions
  - There will be 1 minute of questions after your talk
  - **Trust me, you've put more thought into your work than the audience has**
  - Nobody's going to invalidate your work with their question
  - It's OK not to know
  - Statements like, "that's a good question," buy time to think

Final Report Logistics

- Your final project report is due in 10 days
  - Friday, 06/12/2024 at 11:59PM
  - Late days cannot be used. Unfortunately, no credit will be given for late projects.
- Submit report as PDF on Gradescope by 11:59PM
- Submit code to cs231a-winter2024-teaching@lists.stanford.edu
  - Preferably in the form of a link to a Git repository. Also include link in report.
  - You can also submit cool videos, interactive visualizations, demos, etc.
- Your final write-up should be between 6 - 8 pages
- Use this LaTeX template. Consider:
  - https://www.overleaf.com - free collaborators

Final Project Report Content

- **Abstract**: Concise summary of what your project is about. It should be 200 to 300 words. Mention the general topic area, why your work is novel, perhaps how it relates to the literature, and a brief overview of the results.
- **Introduction:** Provide a concise statement of the problem you're tackling. Succinctly describe the general approach you are taking. Which parts of the problem are you approaching? Possibly provide a brief outline of the paper's content and sections.
- **Background and Related work**: Compare and contrast your work with

Final Project Report Content

- **Technical Content:** These sections detail sub-components of your solution. Be specific and go into detail. You might want to include equations, figures, plots, or tables.
- **Evaluation**: How did you test the algorithms you developed? Start by describing the experiments you're performing and what kind of datasets you're using. How do you measure or evaluate your results? Then show the results of your experiments in detail. Show both quantitative evaluations (show numbers, figures, tables, etc) as well as qualitative results (images, example results, etc). Compare with approaches in the literature or

Final Project Report Content

- **Conclusion:** What have you learned? Make conclusive statements about the issues you faced. Admit any downsides with your solution approach. Suggest future ideas. Quickly outline any work still in progress.
- **References:** This is absolutely necessary. You should include any works that you build upon as well as existing work with similar goals. Cite software packages you use.

Final Project Reports Examples

1. Application of Hough Lines/Canny detection to problem (lane detection)
2. Extension of existing paper (NeRF + Colmap)
3. Novel application of methods (Architectural Neural Sketches)
4. Good example presentation (Fitness Pose Correction)

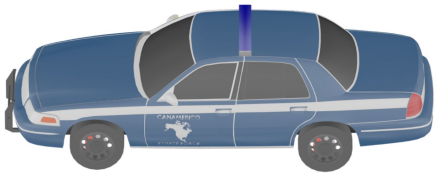
Final Project Templates

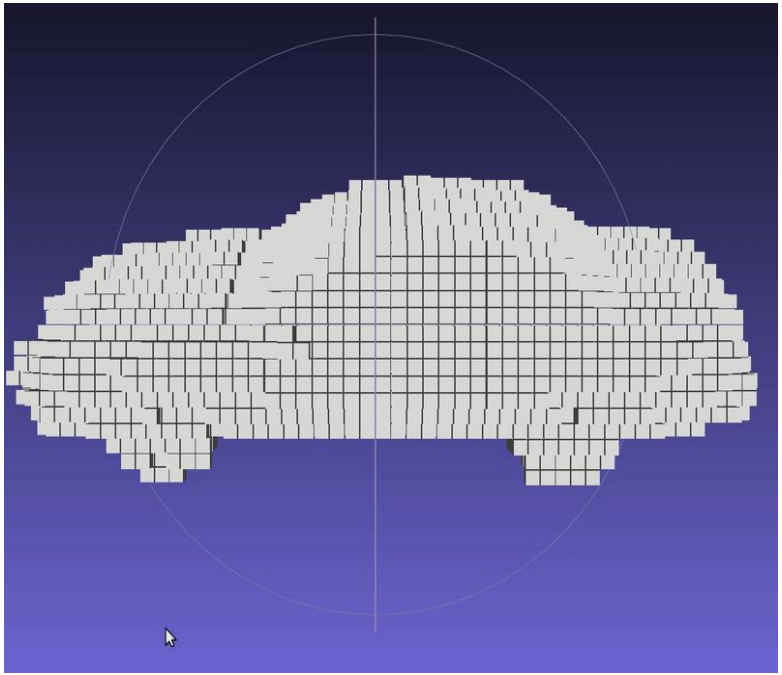
- Poster template: Final Project Template
- Presentation template: (use Google Slides)

Single-View Category-Agnostic 3D Reconstruction

Trevor Standley

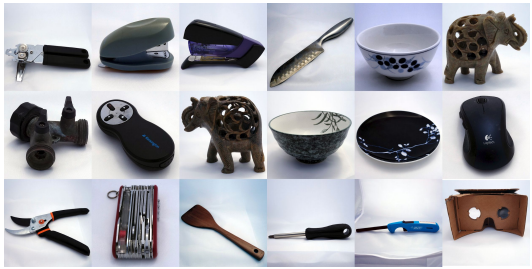
Goal



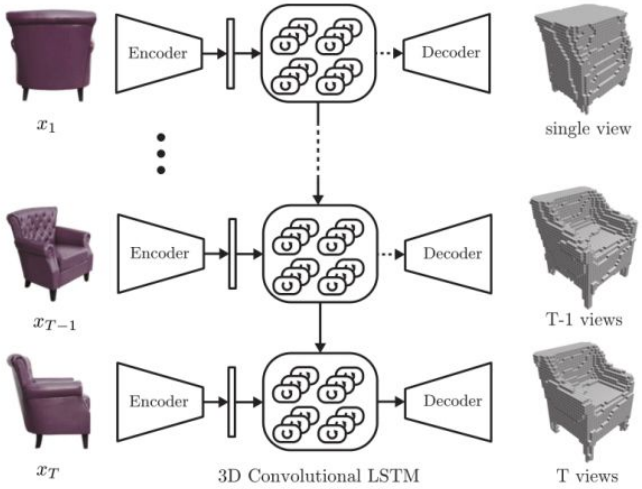


Similar to 3D-R2N2 Except

- Single view only
- Must work for all classes of objects



- Also similar to works that learn to construct a depth map from an image.



3D-R2N2: A Unified Approach for Single and Multi-view 3D Object Reconstruction. Choy et al. 2016

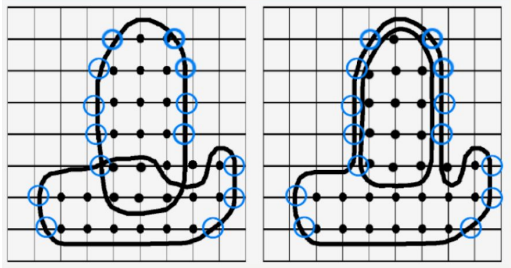
ShapeNet Data

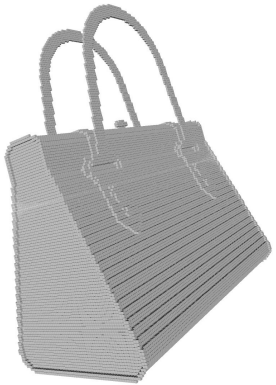
http://shapenet.cs.stanford.edu/

The data

- Render 2d images using Blender

- Wrote custom python/numpy code to convert OBJ files into voxels
  - Ray Stabbing on all three axes then 'AND' the three results.



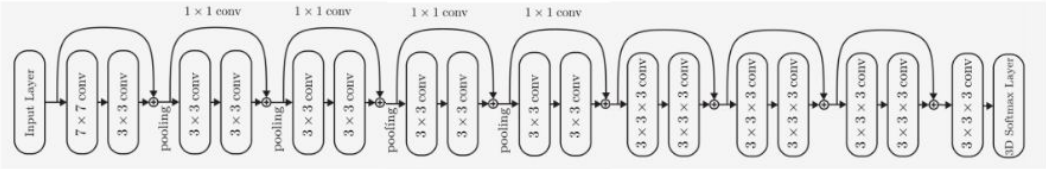


Architecture

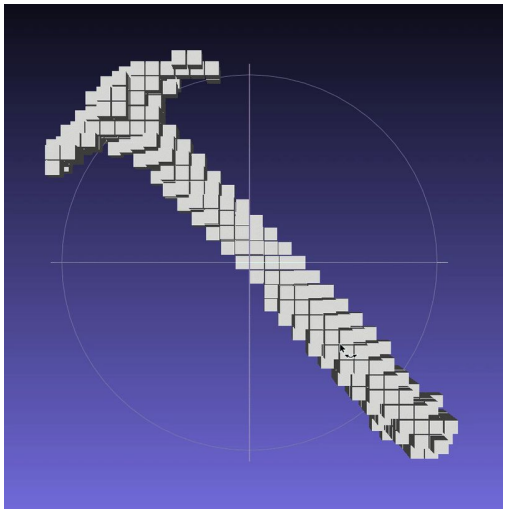
Very simple

- Bunch of 2d residual blocks
- Then a bunch of 3d residual blocks
- ~40M parameters

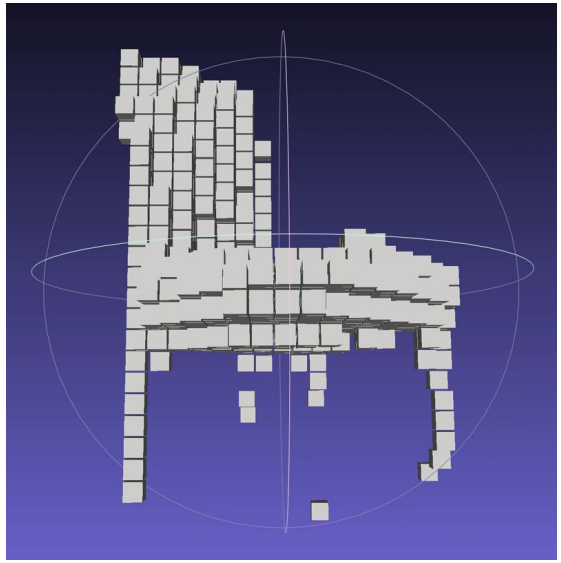
(Approximate)



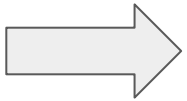
Results

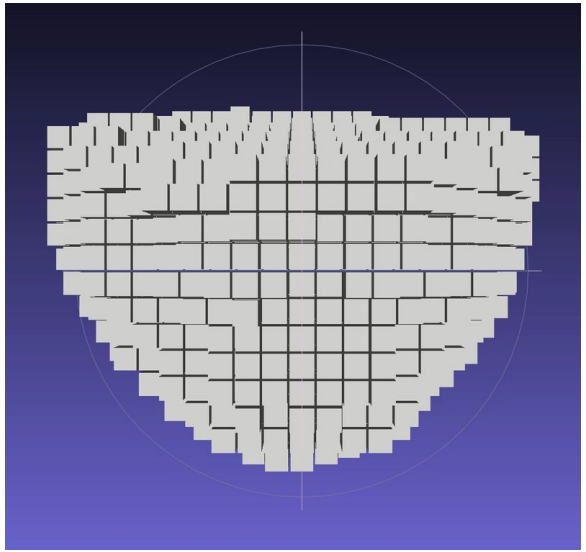


Results



Results





Validation Set Results

Intersection over Union = 0.4145

Challenges

- Wrote my own voxelizer
- Blender is very hard to work with
- Obj files rotated with a different coordinate axis than voxels/images.
- Discovered that Blender has trouble rendering face colors and rendering with backface culling.

Computing the Jacobian

- First-order partial derivative of a vector-valued function with respect to vector-valued input.

$$\mathbf{J} = \begin{bmatrix} \dfrac{\partial \mathbf{f}}{\partial x_1} & \cdots & \dfrac{\partial \mathbf{f}}{\partial x_n} \end{bmatrix} = \begin{bmatrix} \nabla^{\mathrm{T}} f_1 \\ \vdots \\ \nabla^{\mathrm{T}} f_m \end{bmatrix} = \begin{bmatrix} \dfrac{\partial f_1}{\partial x_1} & \cdots & \dfrac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \dfrac{\partial f_m}{\partial x_1} & \cdots & \dfrac{\partial f_m}{\partial x_n} \end{bmatrix}$$

Source: Wikipedia