Prathamesh Arvind Jadhav

Unit-4-Reasoning

Uncertainty: Probabilistic Reasoning in AI (Robotics)

---

1. What is Uncertainty in AI?

- In real-world applications like robotics, environments are **partially observable**, **dynamic**, and **noisy**.
- Due to **sensor errors**, **incomplete knowledge**, or **unpredictable outcomes**, an AI system **cannot always be 100% certain** about the current state or outcome.
- This **uncertainty** must be managed using mathematical techniques.

---

2. Why Uncertainty Matters in Robotics?

Robots rely on:

- **Sensors (e.g., cameras, LiDAR)** to observe the world
- **Actuators** to perform actions
  Both can produce **incomplete or inaccurate data**, making decision-making uncertain.

Example:

- A robot detecting a wall using LiDAR may receive **noisy readings** if the surface is reflective.

---

3. What is Probabilistic Reasoning?

- Probabilistic reasoning is a method of making inferences or decisions based on **probability theory**.
- It **quantifies uncertainty** using probabilities and helps the system **reason under incomplete or noisy information**.

Prathamesh Arvind Jadhav

4. Key Concepts in Probabilistic Reasoning

a) Prior Probability

- Belief before getting new data.

b) Posterior Probability

- Updated belief after observing evidence.

c) **Bayes' Theorem**

- Formula to compute posterior from prior and likelihood:

$$P(H|E) = \frac{P(E|H) \cdot P(H)}{P(E)}$$

where:

- H = hypothesis (e.g., robot is near wall)
- E = evidence (sensor reading)

d) **Joint Probability Distribution**

- Probabilities of all combinations of variables.

e) **Conditional Probability**

- Probability of one event given another occurred.

□ 5. Application in Robotics

a) Localization

- Robot uses probabilistic models to estimate **its position** in an environment.
- Example: **Monte Carlo Localization (MCL)**.

b) Mapping

- SLAM (Simultaneous Localization and Mapping) uses **Bayesian filters** like **Kalman Filters** or **Particle Filters** to update maps based on noisy data.

c) Path Planning

- Robot calculates the **most probable safe path**, considering uncertainties like dynamic obstacles.

d) Sensor Fusion

- Combines data from multiple sensors (e.g., GPS + camera) using probabilities to improve accuracy.

---

6. Example: Self-Driving Car

- Uses probabilistic reasoning to detect obstacles:
  - Camera may detect a blurry shape.
  - Radar may detect an object at a distance.
  - Probabilistic model combines both to decide if it's a pedestrian or not.

---

7. Benefits of Probabilistic Reasoning in AI

- Helps deal with **uncertainty and ambiguity**
- Enables **robust decision-making**
- Improves **reliability and adaptability** in unknown environments

Filtering and Prediction in Probabilistic Reasoning

---

1. Introduction

- In real-world systems like robots, the environment is **uncertain** and **dynamic**.
- AI systems use **probabilistic reasoning** to estimate the state of the world.
- Two fundamental tasks in this reasoning are:
  - **Filtering**
  - **Prediction**

Prathamesh Arvind Jadhav

---

2. What is Filtering?

- **Filtering** is the process of **estimating the current state** of a system, given all the observations **up to the present time**.
- It involves **updating the belief** based on the **latest observation**.

 Example:

- A robot wants to know where it is **right now** using GPS + sensor data.
- It uses filtering to combine all past sensor inputs and current sensor readings to estimate its current location.

---

3. What is Prediction?

- **Prediction** estimates the **future state** of the system, without yet having future observations.
- It uses the current belief and the **transition model** (how the system changes over time) to **predict what will happen** in the next step(s).

 Example:

- A robot predicts where it will be in the next 5 seconds if it continues to move in a certain direction.
- It does this **before actually moving** or seeing new sensor data.

---

4. Key Differences Between Filtering and Prediction

Aspect	Filtering	Prediction
Purpose	Estimate **current state**	Estimate **future state**
Observation usage	Uses **all past and current** observations	Uses **past observations**, no future data
Type of reasoning	**Diagnostic** (what is true now?)	**Prognostic** (what will happen?)
Involves	Bayes update with new	State transition using model

Aspect	Filtering	Prediction
	evidence	

---

5. Application in Robotics

a) Filtering in Robotics:

- Used in **localization** (e.g., where am I now?)
- Example: **Kalman Filter** in self-driving cars to estimate current position using noisy GPS and sensor data.

b) Prediction in Robotics:

- Used in **path planning** or **collision avoidance**
- Predicts where obstacles or the robot itself will be in the near future.

Hidden Markov Models (HMMs) and Kalman Filters

---

1. Introduction

In robotics and AI systems, handling uncertainty is a key challenge. Two commonly used probabilistic models for reasoning under uncertainty are **Hidden Markov Models (HMMs)** and **Kalman Filters**. Both are used to **estimate hidden states** based on observable data but differ in their approaches and applications.

---

2. What are Hidden Markov Models (HMMs)?

- **Hidden Markov Models (HMMs)** are probabilistic models used to represent systems that follow a **Markov process** with hidden states.
- In **HMMs**, the state is **not directly observable** (hidden), but the system produces **observable outputs** that provide indirect information about the state.

Key Components of HMMs:

- **States**: The system's internal condition, which is not directly visible.

- **Observations**: Visible outputs that provide clues about the hidden state.
- **Transition probabilities**: The likelihood of moving from one state to another.
- **Emission probabilities**: The likelihood of observing a certain output given a particular hidden state.
- **Initial probabilities**: The likelihood of starting in a specific state.

Example:

- A robot navigating a maze:
  - **States** could represent the robot's position in the maze.
  - **Observations** could be the sensor data (e.g., distance to walls).
  - **Goal**: Use the sensor data to estimate the robot's position (hidden state).

---

3. Applications of HMMs in Robotics

- **Localization and Mapping**: Robots use HMMs to estimate their position based on sensor readings and environmental states.
- **Speech Recognition**: Used to recognize spoken words by modeling the sequence of speech features as a hidden process.
- **Gesture Recognition**: HMMs help identify a sequence of movements or gestures performed by a user or a robot.

---

4. What are Kalman Filters?

- **Kalman Filters** are a type of **recursive estimator** used to estimate the state of a system over time.
- It is widely used in applications where the system state is subject to **noise** and **uncertainty**.
- Kalman Filters estimate the **hidden state** of a linear dynamic system based on noisy measurements and prior knowledge about the system's dynamics.

Key Features of Kalman Filters:

- **State prediction**: The filter uses the previous state to predict the current state.

- **Measurement update**: It combines the predicted state with the current measurement to correct the state estimate.
- **Noise modeling**: Kalman filters model both the process noise (system dynamics) and measurement noise (sensor errors).

Example:

- A robot using **GPS** and **accelerometer** to track its position:
  - The **GPS data** may have noise, and the **accelerometer** data can drift over time.
  - The **Kalman Filter** combines both sensor readings to give a more accurate position estimate.

---

5. Applications of Kalman Filters in Robotics

- **Navigation and Localization**: A robot uses Kalman Filters to fuse data from multiple sensors (e.g., GPS, IMU) to improve its position estimate.
- **Path Planning**: Used to predict the future state of a robot and its surroundings, aiding in collision avoidance.
- **Tracking**: In autonomous vehicles, Kalman Filters are used to track moving objects and predict their future positions.

---

6. Differences Between HMMs and Kalman Filters

Aspect	Hidden Markov Models (HMMs)	Kalman Filters
State Model	Discrete hidden states	Continuous hidden states
System Dynamics	Typically for discrete systems	Used for linear systems with Gaussian noise
Prediction Method	Based on state transition probabilities	Based on linear state equations and noise models
Observations	Emission probabilities of outputs	Direct measurements used to correct state estimates

---

7. When to Use HMMs vs Kalman Filters

- **HMMs** are best when the system has **discrete, hidden states** (e.g., robotic actions, speech recognition).
- **Kalman Filters** are ideal for **continuous state systems** where the state evolves linearly over time and measurements are noisy.

Dynamic Bayesian Networks (DBNs)

---

1. Introduction to Dynamic Bayesian Networks (DBNs)

- **Dynamic Bayesian Networks (DBNs)** are an extension of **Bayesian Networks** designed to model **temporal processes**.
- Unlike traditional **Bayesian Networks**, which model static relationships, **DBNs** handle **sequences of variables** where the state of the system at a given time depends on both its previous state and some external evidence.
- DBNs allow reasoning over **time-series data** and **sequential decision-making** tasks, making them ideal for applications like robotics, speech recognition, and time-series prediction.

---

2. Structure of Dynamic Bayesian Networks

- A **DBN** consists of two key components:
    1. **Temporal Structure**: Represents how variables evolve over time.
    2. **Conditional Independence**: Nodes at any given time are conditionally independent of each other, given their parents.
- **DBNs** are modeled as **a chain of Bayesian Networks**, where each network corresponds to a snapshot in time.
    - **Nodes** represent **variables** at different time steps (e.g., position of a robot at time ttt, velocity at time ttt).
    - **Edges** represent **conditional dependencies** between variables across time.

---

3. Components of DBNs

1. **State Variables**: Variables that describe the system's state at a given time (e.g., robot's position, velocity).

2. **Observations**: Variables that are observed at each time step (e.g., sensor readings).
3. **Transition Model**: Defines how the state evolves over time. It is a probabilistic model that relates the state at time t−1t-1t−1 to the state at time ttt.
4. **Emission Model**: Defines how the observations at time ttt are generated from the state at time ttt.

---

4. Key Features of DBNs

- **Temporal Evolution**: Unlike static Bayesian Networks, DBNs are used to model how the system evolves over time.
- **Markov Assumption**: DBNs generally rely on the **Markov property**, meaning the current state depends only on the previous state, not on past history beyond the immediate predecessor.
- **Efficient Inference**: Inference in DBNs is usually done by **message-passing algorithms** or using methods like **variational inference** to update beliefs about the system's state as new observations come in.

---

5. Applications of DBNs in Robotics

- **Robot Localization**: DBNs can model a robot's position over time, updating its belief about its position as it receives new sensor readings.

Example:

- A robot uses **DBNs** to estimate its location based on noisy sensor data (e.g., laser rangefinders, GPS).
- At time ttt, the **robot's state** (e.g., position and velocity) is updated using the **transition model**, and the robot's **sensor readings** update the belief about its current state through the **emission model**.
- **Robotic Motion Planning**: DBNs can predict a robot's movement over a series of steps, considering uncertainties in motor commands and sensor noise.
- **Tracking Dynamic Objects**: DBNs are used in **object tracking** where the position of objects is inferred over time with observations being noisy.

---

6. Inference in DBNs

- **Inference** involves computing the **posterior distributions** of the system's state, given the observed evidence.
    - At each time step, the **posterior** is updated by combining the **prior belief** from the transition model and the new **evidence** from the observation.

Types of Inference Methods:

1. **Exact Inference**: Solves the problem using algorithms like **variable elimination** or **belief propagation** (typically infeasible for large-scale problems).
2. **Approximate Inference**: Methods like **Monte Carlo Markov Chains (MCMC)** or **Particle Filters** are used to approximate the posterior when exact inference is computationally expensive.

---

7. Advantages of DBNs

- **Dynamic Modeling**: Suitable for systems that evolve over time, such as robots navigating in dynamic environments.
- **Uncertainty Handling**: DBNs naturally incorporate uncertainty in state transitions and observations, making them robust in real-world applications.
- **Scalability**: Can model complex systems with large numbers of variables across multiple time steps.
- **Versatility**: Used in a wide range of fields such as **robotics**, **speech recognition**, and **financial forecasting**.

---

8. Challenges in DBNs

- **Complexity**: DBNs can become computationally expensive for large-scale systems due to the large number of dependencies and time steps.
- **Data Requirements**: DBNs require large amounts of **data** to accurately learn the transition and emission models, which can be challenging in some real-world scenarios.

- **Inference Difficulty**: Exact inference can be infeasible, and even approximate inference methods may struggle with large systems.

---

9. Example Use Case: Robot Navigation

- **Scenario**: A robot is navigating a warehouse and needs to track its position using sensors.
  - **State Variables**: The robot's position (x, y) and velocity (v) over time.
  - **Transition Model**: Models how the robot's position changes based on its movement commands (e.g., moving forward with some speed).
  - **Emission Model**: Models how the robot's sensor readings (e.g., from cameras or LIDAR) depend on its position.
  - **DBN** will allow the robot to continuously update its belief about its position and adjust its navigation decisions as it moves through the environment.

Speech Recognition and Making Decisions in AI

---

1. What is Speech Recognition?

- **Speech Recognition** is the ability of a computer or machine to **convert spoken language into text**.
- It is a sub-field of **Natural Language Processing (NLP)** and plays a vital role in **human-computer interaction**.
- Also known as **Automatic Speech Recognition (ASR)** or **Speech-to-Text**.

---

2. How Speech Recognition Works

1. **Audio Input**: The user speaks into a microphone.
2. **Signal Processing**: The system captures the sound and converts it into a **digital waveform**.
3. **Feature Extraction**: Extracts key audio features like pitch, frequency, etc.

4. **Acoustic Modeling**: Uses models like **Hidden Markov Models (HMMs)** or **Deep Neural Networks (DNNs)** to match audio features to phonemes (basic units of sound).
5. **Language Modeling**: Predicts the most likely word sequence using **grammar rules** or **probabilities**.
6. **Output Generation**: Produces the most likely text form of the spoken input.

---

3. Applications of Speech Recognition

- **Voice Assistants**: Siri, Alexa, Google Assistant.
- **Transcription Tools**: Converting audio lectures, interviews into text.
- **Voice Commands in Devices**: Smart TVs, home automation, cars.
- **Accessibility**: Helping visually or physically impaired users interact with technology.

---

4. Challenges in Speech Recognition

- **Accents and Dialects**: Variability in pronunciation affects accuracy.
- **Background Noise**: Makes it hard to extract clear speech signals.
- **Homophones**: Words that sound the same but have different meanings (e.g., “write” vs. “right”).
- **Context Understanding**: Requires AI to understand the context for accurate recognition.

□ Making Decisions in AI

---

5. What is Decision Making in AI?

- **Decision making** in AI refers to the process of **choosing the best action** from a set of alternatives to achieve a goal.
- Involves analyzing data, evaluating possible outcomes, and selecting the most optimal choice using logic or probabilities.

- Often implemented in **autonomous systems** like self-driving cars, robots, and AI agents in games.

---

6. Techniques Used in AI for Decision Making

1. **Rule-Based Systems**: Use "if-then" logic rules.
2. **Decision Trees**: Break down decisions into a tree-like model of choices.
3. **Bayesian Networks**: Make probabilistic decisions under uncertainty.
4. **Reinforcement Learning**: Learn optimal actions through rewards and penalties.
5. **Utility Theory**: Chooses actions that maximize expected utility (benefit).

---

7. Examples of AI Decision Making

- **Robotics**: Choosing paths or actions based on environment sensors.
- **Self-driving cars**: Deciding when to stop, turn, or accelerate.
- **Healthcare AI**: Recommending treatments based on patient data.
- **Finance**: Stock trading bots making buy/sell decisions.

□ 8. Challenges in AI Decision Making

- **Uncertainty**: AI must often act with incomplete or noisy data.
- **Real-time Constraints**: Decisions must be made within milliseconds in critical systems.
- **Bias and Fairness**: Decision systems can unintentionally favor or harm certain groups if not trained fairly.

---

9. Integration: Speech Recognition + Decision Making

- Many modern AI systems combine **speech recognition** with **decision-making**.

  □ **Example:**

- A voice assistant hears the command: “Book me a cab.”
- **Speech Recognition** converts voice to text.
- **Decision-Making Module** interprets the intent and decides which app to use, what time, and destination.
- The AI system then executes the booking.