**Name: Paras Topwal (2022A7PS1243G)**

**Simultaneous Localization and Mapping (SLAM)** is a fundamental part of autonomous system which allows the robot to create a map of the unknown surroundings while simultaneously locating its position. It takes input from **cameras, LIDAR** (Light Detection and Ranging-it is similar to SONAR, just that it uses light pulses instead of sound waves), and **IMUs** (Inertial Measurement Units: used to measure robot’s motion and orientation and consists of accelerometer, gyroscope and magnetometer), and creates a map of the surroundings and its own location.

It consists of two steps:

1. **Mapping**: As the robot moves, it gathers sensor data and extracts features from its surroundings which helps it build a map of the environment (2D or 3D).
2. **Localization**: Here also the robot uses sensor data to estimate its position within the map. It compares the sensor data to the mapped features and accordingly adjust the robot’s pose (it includes position and orientation). We compare the data from sensor to the amp because the sensor data often is noisy and inaccurate).

The different types of SLAM algorithms include:

1. **EKF SLAM (Extended Kalman Filter** SLAM): It employs Extended Karman Filter which is a recursive, non-linear estimation algorithm which estimates the current state based on noisy sensor data while considering uncertainty in both measurements and the state. It is modified to assume **linearized sensor data** which works well in small-scale models but does not work with non-linearities in larger and complex models.
2. **Particle Filter SLAM (Monte Carlo SLAM):** It uses a collection of **particles** in such a way that it propagates them through the sensor data to estimate the robot's path and its surroundings. It handles complex and large systems **better than EFK algorithm** but it is computationally demanding due to requirement of large number of particles.
3. **Graph-based SLAM:** It treats the SLAM as a graph where **nodes** represent robot poses (**pose node**) and map features (**landmark** **node**) and **edges** (connects the two nodes) represent constraints from sensor measurements. To minimize these constraints, it uses optimizing techniques like **Bundle** **Adjustment** (aims to minimize the error between predicted and actual measurements) that offers flexibility for large-scale and complex environments. This also uses techniques such as **pose graph optimization** (keeps landmark fixed and increases the accuracy of robot’s trajectory) and **keyframe-based SLAM** (it optimizes a subset of poses called keyframes).