1. Kinematics (object detection, pose estimation, camera calibration)

Kinematika adalah studi tentang gerakan benda tanpa adanya pertimbangan mengenai gaya yang menyebabkan gerakan tersebut. Kinematika dapat dibagi menjadi:
Forward Kinematics (FK): menghitung posisi orientasi ujung robot berdasarkan posisi setiap joint
Inverse Kinematics (IK): mencari posisi joint yang diperlukan untuk mencapai posisi orientasi ujung robot yang diinginkan
Object detection berfungsi mengklasifikasikan objek dalam suatu gambar atau video umumnya
Pose estimation berfungsi memperkirakan posisi (x,y.z) dan rotasi dari objek dalam 3D
Camera calibration berfungsi mengoreksi distorsi serta mencari parameter intrinsik dan ekstrinsik

2. ADRC (Active Disturbance Rejection Control)

ADRC adalah teknik kontrol modern yang dirancang untuk mengatasi gangguan eksternal yang tidak diketahui. ADRC fleksibel terhadap perubahan parameter sistem dan gangguan eksternal.
ADRC terdiri dari beberapa komponen utama, yaitu:
-Extended State Observer (ESO): berfungsi memperkirakan gangguan eksternal yang tidak diketahui secara real time.
-Nonlinear State Error Feedback (NLSEF): mengatur kontrol berdasarkan sinyal eror
-Total Disturbance Rejection: memperkirakan dan mengkompensasi gangguan berdasarkan pengamatan ESO`
Keunggulan ADRC adalah efektif mengatasi gangguan eksternal serta tidak bergantung pada model presisi sistem, sehingga cocok untuk suatu model kompleks.

3. PID (Proportional-Integral-Derivative) control algorithms

PID mengontrol keluaran sistem berdasarkan error antara input yang diinginkan dan keluaran aktual, komponennya terdiri dari 3 bagian, yaitu:
-Proportional: Respons kontrol sebanding dengan besar error
-Integral: Menghitung akumulasi error dengan pendekatan integral sehingga lebih akurat
-Derivative: Memprediksi perubahan error ke depan untuk melakukan pencegahan
Keunggulan PID adalah mudah diterapkan dalam sistem kontrol sederhana, fleksibel, dan membantu menstabilkan sistem yang tidak stabil.

4. A* (A star) algorithm

A* berfungsi untuk menemukan jalur terpendek antara dua titik dalam peta sambil menghindari rintangan.
A* menggunakan dua fungsi untuk mengevaluasi node (titik di peta)
-Cost so far (g(n)): biaya perjalanan titik awal hingga node n
-Heurestic (h(n)): perkiraan biaya dari node n ke tujuan (memakai jarak Euclidean atau Manhattan)
Total biaya f(n)=g(n)+h(n)
Langkah-langkah A*:
Mulai dari node awal, tambahkan ke open list.
Dari node yang ada di open list, pilih node dengan biaya f(n) terendah.
Jika node tersebut adalah tujuan, maka algoritma selesai, dan jalur ditemukan.
Jika tidak, pindahkan node ke closed list dan tambahkan tetangga node tersebut ke open list jika tetangga belum pernah dikunjungi.
Ulangi hingga tujuan tercapai atau tidak ada node tersisa di open list.
Heuristik dalam A*:

Pemilihan heuristik sangat penting dalam A*. Beberapa contoh heuristik:

Manhattan Distance: Untuk peta grid, jarak total yang harus ditempuh secara horizontal dan vertikal.
Euclidean Distance: Jarak lurus (garis terpendek) dari node ke tujuan.
Octile Distance: Untuk grid di mana gerakan diagonal diperbolehkan, menggunakan kombinasi jarak diagonal dan manhattan.
Keunggulan A*: Optimal untuk mengurangi biaya dan efisiensi waktu