1. Kinematika adalah algoritma yang berhubungan dengan konektivitas rantai kinematik posisi, kecepatan, dan percepatan setiap frame pada sistem robotika untuk merencanakan dan mengendalikan Gerakan serta menghitung gaya dan torsi aktuator. Algoritma ini terbagi menjadi dua, yaitu kinematika maju dan kinematika terbalik. Kinematika maju menentukan parameter sambungan dan menghitung konfigurasi rantai. Sedangkan, Kinematika invers menentukan lokasi end-effector dan menghitung sudut joint yang terkait.

2. Algoritma ADRC adalah metode kontrol yang tidak menggunakan model dan digunakan untuk mendesign control untuk sistem dengan gerakan yang tidak diketahui dan gangguan eksternal. Penggunaan algoritma ini berfungsi untuk mengetahui perilaku sistem sehingga dapat membuat kontrol yang dapat mengatasi gangguan. Untuk mencapai kekokohan, ADRC (Active Disturbance Rejection Control) didasarkan pada perluasan model sistem dengan variabel keadaan tambahan dan fiktif yang mewakili segala sesuatu yang tidak disertakan pengguna dalam deskripsi matematis dari sistem dasar yang akan dikendalikan. Fitur penolakan gangguan ini memungkinkan pengguna untuk memperlakukan sistem yang dipertimbangkan dengan model yang lebih sederhana sejauh efek negatif dari ketidakpastian pemodelan dikompensasi secara real-time. Akibatnya, operator tidak memerlukan deskripsi analitis yang tepat dari sistem dasar; seseorang dapat memodelkan bagian dinamika yang tidak diketahui sebagai gangguan internal dalam sistem dasar.

3. Proportional-integral-derivative controller (PID) adalah mekanisme loop kontrol berbasis umpan balik yang umum digunakan untuk mengelola mesin dan proses yang memerlukan kontrol kontinu dan penyesuaian otomatis. Pengendali PID secara otomatis membandingkan nilai target yang diinginkan (setpoint atau SP) dengan nilai aktual dari sistem (variabel proses atau PV). Kemudian, pengendali PID secara otomatis menerapkan tindakan korektif untuk membawa PV (variabel proses) ke nilai yang sama dengan SP (setpoint) menggunakan tiga metode: Komponen proporsional (P) merespons nilai kesalahan saat ini dengan menghasilkan keluaran yang sebanding dengan besarnya kesalahan. Ini memberikan koreksi langsung berdasarkan seberapa jauh sistem dari setpoint yang diinginkan. Komponen integral (I), pada gilirannya, mempertimbangkan jumlah kumulatif dari kesalahan-kesalahan sebelumnya untuk menangani kesalahan steady-state yang masih ada seiring waktu, sehingga menghilangkan ketidaksesuaian yang berlarut-larut. Terakhir, komponen derivatif (D) memprediksi kesalahan di masa depan dengan menilai laju perubahan kesalahan, yang membantu mengurangi overshoot dan meningkatkan stabilitas sistem, terutama ketika sistem mengalami perubahan cepat. Pengendali PID mengurangi kemungkinan kesalahan manusia dan meningkatkan otomatisasi.

4. A* adalah algoritma traversal grafik dan pencarian jalur. Diberikan sebuah grafik berbobot, sebuah node sumber, dan sebuah node tujuan, algoritma ini menemukan jalur terpendek (berdasarkan bobot yang diberikan) dari sumber ke tujuan. Algoritma ini bekerja dengan mempertahankan sebuah pohon jalur yang berasal dari node awal dan memperluas jalur tersebut satu sisi pada satu waktu hingga node tujuan tercapai. Pada setiap iterasi dari loop utamanya, A* perlu menentukan jalur mana yang harus diperluas. Ia melakukannya berdasarkan biaya jalur dan perkiraan biaya yang dibutuhkan untuk memperluas jalur tersebut hingga mencapai tujuan.

Sumber :
https://motion.cs.illinois.edu/RoboticSystems/Kinematics.html
https://www.sciencedirect.com/topics/engineering/active-disturbance-rejection-control
https://www.omega.com/en-us/resources/pid-controllers#:~:text=A%20PID%20(Proportional%20%E2%80%93%20Integral%20%E2%80%93,variables%20in%20industrial%20control%20systems.
https://www.geeksforgeeks.org/a-search-algorithm/