Opciones de diseño e implementación de sistemas robotizados

El diseño e implementación de un sistema robotizado no es simplemente ensamblar componentes, sino que implica una **planificación completa del hardware y del software**, considerando su entorno, función, escalabilidad y mantenibilidad.

Los factores clave incluyen:

- El entorno en el que operará el robot (por ejemplo, si es corrosivo o extremo).
- El tipo de tareas que realizará (coger y colocar, soldar, desplazarse, inspeccionar, etc.).
- Las restricciones físicas y lógicas del sistema.
- Aspectos éticos y de seguridad, especialmente en contextos industriales.

Cada robot debe diseñarse con base en estos elementos para asegurar eficiencia, seguridad y sostenibilidad.

Clasificación de los robots según su tipología física

Los robots pueden clasificarse por su estructura física, lo cual influye directamente en su movilidad, precisión y aplicaciones. Aquí te presento las principales tipologías:

1. Robot cartesiano

Se mueve en líneas rectas a lo largo de los ejes X, Y y Z. Su movimiento es sencillo y controlado. Poseen una gran precisión en trayectorias lineales, ideal para tareas de ensamblaje o corte.

2. Robot Multi-DOF (múltiples grados de libertad)

Tiene más articulaciones, lo que le permite orientarse en múltiples direcciones, pueden alcanzar cualquier punto con cualquier orientación, ideal para tareas complejas como soldadura o pintura.

3. Robot SCARA

Diseñado para trabajar en planos horizontales, con 3 ejes lineales y uno rotacional. Cuenta con rapidez y eficiencia en tareas repetitivas como “pick and place”.

4. Robot DELTA o tipo araña

Estructura paralela con brazos articulados en forma de paralelogramo. Su principal característica es una gran velocidad y precisión en tareas ligeras, como clasificación en líneas de producción.

¿Qué es ROS (Robot Operating System) y por qué es tan importante?

ROS, o **Robot Operating System**, es una infraestructura de software que permite a los desarrolladores construir sistemas robóticos complejos y modulares. Aunque su nombre diga "sistema operativo", en realidad es un **middleware**, es decir, una capa de software que actúa como intermediaria entre el hardware del robot y sus aplicaciones.

¿Qué lo hace especial?

- **Arquitectura por nodos:** permite dividir el sistema robótico en partes independientes (por ejemplo, uno para visión, otro para movimiento, otro para sensores), lo que facilita el desarrollo y la depuración.
- **Interoperabilidad entre lenguajes:** permite mezclar lenguajes como Python y C++ sin problema.
- **Comunidad activa:** ROS es open source y cuenta con miles de paquetes y librerías ya desarrolladas para visión, navegación, SLAM, control de brazos, etc.
- **Integración con Jetson y otros sistemas embebidos:** como se menciona en el documento, placas como Jetson Nano o Xavier usan ROS para gestionar sensores, motores y cámaras de forma eficiente.

Es importante en la robótica moderna porque permite acelerar el desarrollo, probar algoritmos en simuladores (como Gazebo o Webots), y facilita el paso de prototipo a implementación real. Además, al usarlo como estándar, permite que desarrolladores de diferentes partes del mundo colaboren usando los mismos “bloques”.