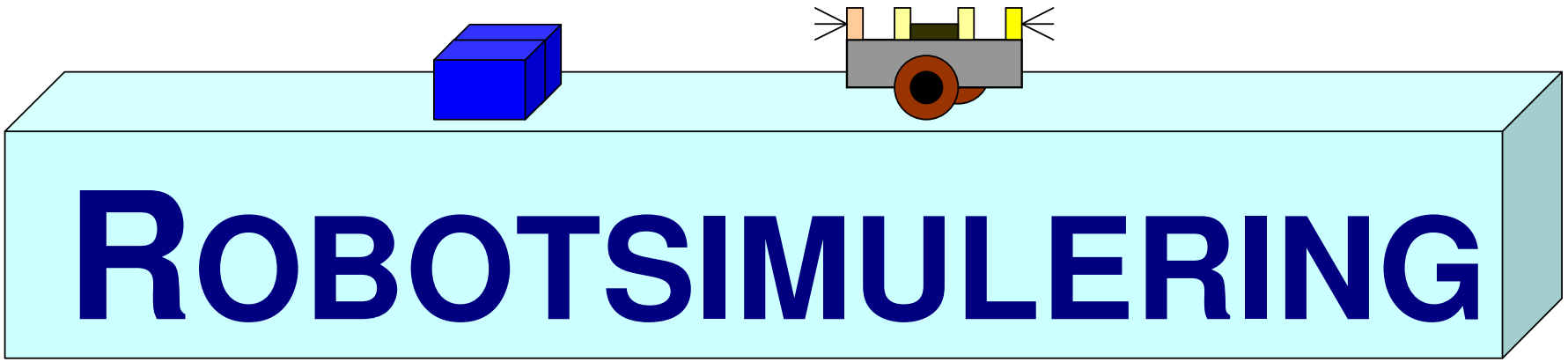


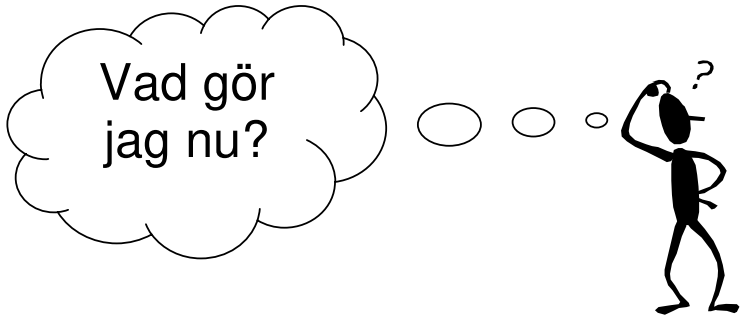
Ett samarbete mellan Tim portnoff och Henrik Lernmark

Robotinlärning

En robots nuvarande problemställning är för tillfället

- **Var är jag, var ska jag och vad ser jag?**
- **Hur skall jag kombinera detta till något vettigt?**

Robotinlärning är en adaptiv process som kombinerar indata från sensorer till utdata till effektorer och som förbättrar robotens prestation allteftersom tiden går



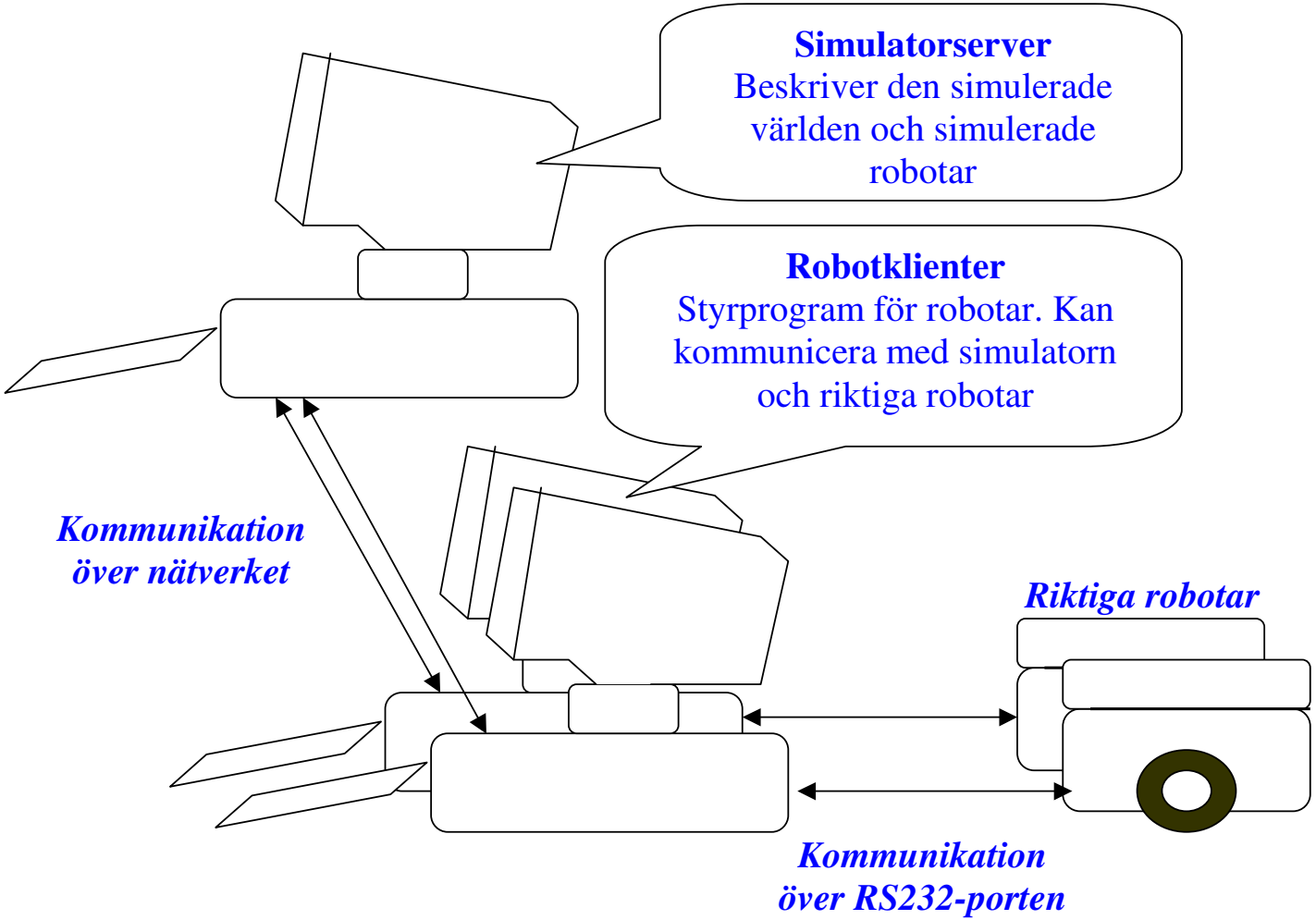
Varför Robotsimulering?

• **Kostnadseffektivt**	**När väl simulatorn är byggd kan många använda den**
• **Tillåter misstag**	**Inga dyra robotdelar kan gå sönder**
• **Ger en första förståelse**	**Simulatorn utreder problematiken kring robotstyrning och inlärning**
• **Snabbare inlärning**	**Inlärningsalgoritmer kan lära sig i simulerad tid**

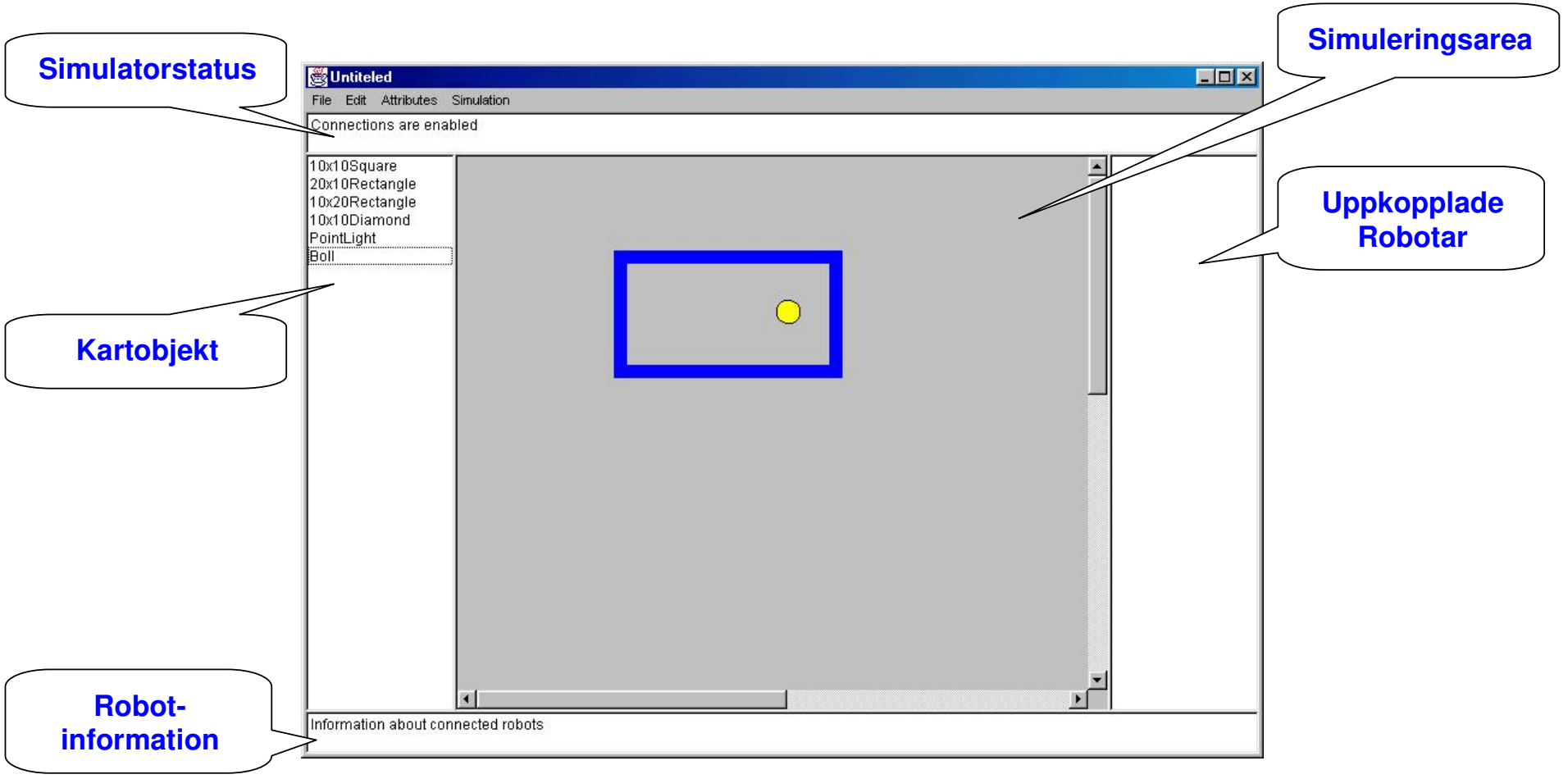
Vår Simulator är

- **Objektorienterad**
- **Skriven i java**
- **Arbetar i realtid**
- **Skalbar över nätverket**
- **Flexibel (Inlärningsalgoritmen kopplas till roboten)**
- **Utbyggbar med nya komponenter**

Simulatorns delar och kommunikation

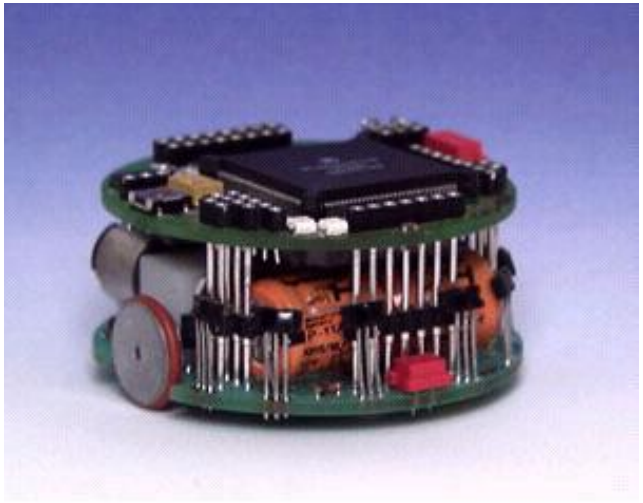


Grafiskt Gränssnitt



Roboten Khepera

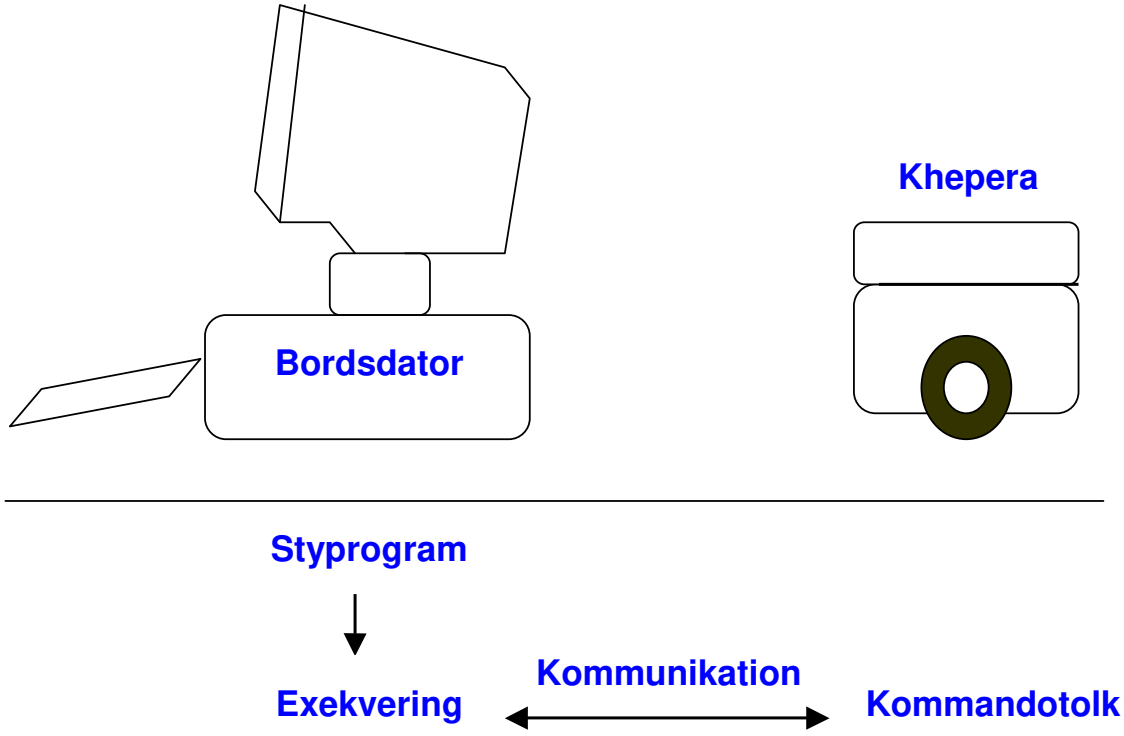
Konstruerad av Labratoire de Microinformatique



- **Flera kommunikationsalternativ**
- **Relativt Billig**
- **Bra utbyggnadsmöjligheter**
- **Liten**
- **68030 processor**

Kommunikation med Kheperan

Styrprogrammet kommunicerar med den inbyggda kommandotolken i Kheperan



Sensorer

Sensorer mäter någon aspekt av omgivningen och omvandlar dessa till ett eller flera reella tal

- **Fotosensor** **Mäter ljus från lampor**
- **Avståndssensor** **Mäter avstånd till objekt**
- **Närhetssensor** **Mäter reflekterat ljus från närbelägna objekt**
- **Kompass** **Bedömer riktningen på den robot den sitter på**
- **GPS** **Bedömer positionen av roboten den sitter på**
- **Kartograf** **Ger en bild av hur omgivningen ser ut**

Effektorer

Effektorer påverkar omgivningen på något sätt

- **Tvåhjulsmotor** — **Ger roboten den sitter på en hastighet och en riktning med programmerbar acceleration och maxhastighet**
- **Lampa** — **Genererar ljus**
- **Lysdiod** — **Genererar lite mindre ljus**

Exempel på en sensor och en effektor

Fotosensor mäter ljus från lampor i omgivningen enligt:

$$sensorvärd\ e = \sqrt{\frac{\sum_{\forall lampor}\left[\left(1 - ljusstyrka \times \frac{\Delta vinkel}{\pi} \times \prod_{\forall objekt\ _i\ _vägen \in synliga} 1/2\right)^2 \times \left(1 - \frac{avstånd}{sensordjup}\right)\right]^2}{antal\ _lampor}}$$

Tvåhjulsmotorn påverkar robotens hastighet och riktning enligt:

$$Rotation = \arctan\left(\frac{hjulavst}{d_0 - d_1}\right) \qquad Hastighet = \frac{(d_0 + d_1)}{2}$$

Med en maximal hastighetsändring mellan två updateringar som avgörs av accelerationen

Kartobjekt

Kartobjekt är de grundläggande byggklossarna för att skapa en bana i robotsimulatorn.

Ett kartobjekt har:

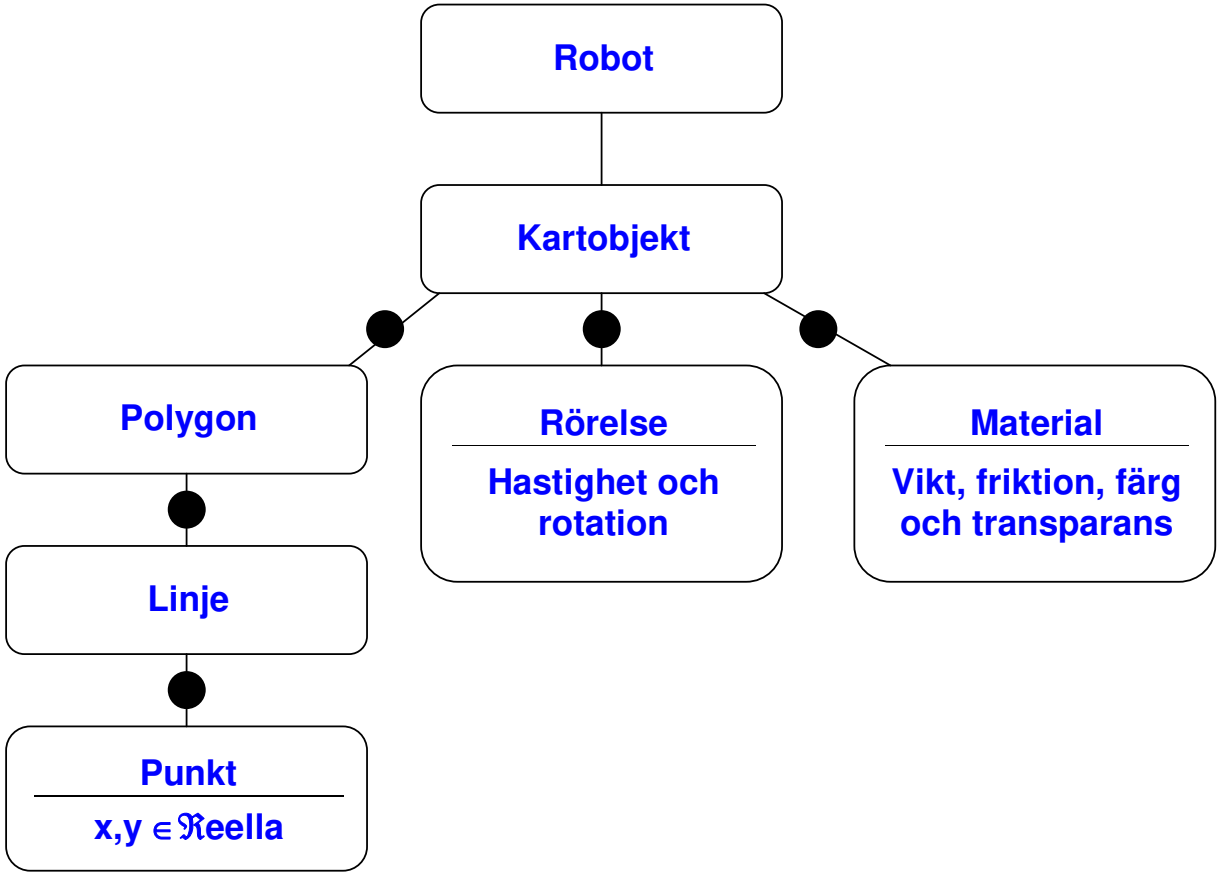
- **ett material**
- **en rörelse**
- **en polygon**

Ett kartobjekt kan ha:

- **Ett annat kartobjekt kopplat till sig**
- **En eller flera effektorer**
- **En eller flera sensorer**

Exempel på kartobjekt är lampor, bollar och väggbitar

Aggregationsmodell från klassen robot



Brister i simulatorn

- Simulatorn fungerar bra men är lite långsam när det finns många objekt på skärmen
- Kollisonshantering är inte komplett och hanterar ej rotationsändringar
- Många sensorvärden är uträknade från förenklade algoritmer
- Simulatorn kan bara köra i realtid
- Man kan inte konstruera nya robotar utan att kunna java
- Vissa av Kheperans kommandon är inte implementerade
- Kommunikationen över RS-232 med den riktiga Kheperan är inte gjord än

Positivt med Simulatorn

- Programmet har ett grafiskt gränssnitt som gör det lätt för användaren att testa en robot
- Det är lätt att skapa ett nytt styrprogram (om man kan Java)
- Det är möjligt att definiera en ny robot genom att bara skapa ett par nya klasser.
- Programmet kan lätt utökas i funktionalitet genom att definiera nya sensorer eller effektorer
- Nybörjare kan koppla på extra sensorer som underlättar styrning och inlärning (ex. GPS och Kompass)
- Skalbarhet genom att utnyttja flera datorer möjliggör större projekt

Tack!