Multi-Robot Systems & Multi-Agent Systems

Marisela Forsythe

First Edition, 2012

ISBN 978-81-323-3631-0

Published by:
University Publications
4735/22 Prakashdeep Bldg,
Ansari Road, Darya Ganj,
Delhi - 110002
Email: info@wtbooks.com

Table of Contents

Chapter- 1

Autonomous Logistics

Autonomous logistics describes systems that provide unmanned, autonomous transfer of equipment, baggage, people, information or resources from point-to-point with minimal intervention. Autonomous logistics is a new area being researched and currently there are few papers on the topic, with even fewer systems developed or deployed.

Autonomous logistics vehicles

There are several subclasses of autonomous logistics vehicles:

Ground autonomous logistics

Based on Unmanned ground vehicle technology, a large autonomous logistics tracked carrier, which can be deployed in a tropical forest for day and night, has been developed .

Another example is the TerraMax autonomous truck based on Oshkosh’s Medium Tactical Vehicle Replacement (MTVR) defense truck platform. Most recently, TerraMax competed in the 2007 Darpa Urban Challenge. The MTVR was designed for the US Marine Corps with a 70% off-road mission profile. TerraMax's unmanned ground vehicle kit does not interfere with the conventional operation of the vehicle. A robust sensor suite allows for 360-degree situational awareness around TerraMax. Elements of the autonomous navigation kit could be used to enhance driver awareness. The complete kit could be used in applications such as snow removal on airport runways.

Unmanned ground vehicle

A Gladiator Tactical Unmanned Ground Vehicle

Unmanned ground vehicles (**UGV**) are robotic platforms that are used as an extension of human capability. This type of robot is generally capable of operating outdoors and over a wide variety of terrain, functioning in place of humans.

UGVs have counterparts in aerial warfare (unmanned aerial vehicle) and naval warfare (remotely operated underwater vehicles). Unmanned robotics are actively being developed for both civilian and military use to perform dull, dirty, and dangerous activities. Some UGVs are employed in War in Iraq.

There are two general classes of unmanned ground vehicles: Teleoperated and Autonomous.

An **unmanned ground combat vehicle** (**UGCV**) is an autonomous, all terrain unmanned ground vehicle designed for combat.

A US Army Multifunctional Utility/Logistics and Equipment (MULE)

Teleoperated UGV

A teleoperated UGV is a vehicle that is controlled by a human operator at a remote location via a communications link. All cognitive processes are provided by the operator based upon sensory feedback from either line-of-sight visual observation or remote sensory input such as video cameras. A basic example of the principles of teleoperation would be a toy remote control car. Each of the vehicles are unmanned and controlled at a distance via a wired or wireless connection while the user provides all control based upon observed performance of the vehicle.

There are a wide variety of teleoperated UGVs in use today. Predominantly these vehicle are used to replace humans in hazardous situations. Examples are explosives and bomb disabling vehicles.

Foster-Miller TALON SWORDS units (military robot) equipped with various weaponry.

Some examples of teleoperated UGV technology are:

- Frontline Robotics Teleoperated UGV (TUGV)
- Gladiator Tactical Unmanned Ground Vehicle (used by the United States Marine Corps)
- iRobot PackBot
- Foster-Miller TALON
- Remotec ANDROS F6A
- Autonomous Solutions Chaos
- Mesa Associates Tactical Integrated Light-Force Deployment Assembly (MATILDA)
- Vecna Robotics Battlefield Extraction-Assist Robot (BEAR)
- G-NIUS Autonomous Unmanned Ground Vehicles (Israel Aerospace Industries/Elbit Systems joint venture) Guardium
- Robowatch ASENDRO
- Kairos Autonomi - Pronto4 System
- Ripsaw MS1
- DRDO Daksh
- VIPeR
- DOK-ING mine clearing, firefighting, and underground mining UGV's
- MacroUSA Armadillo V2 Micro UGV (MUGV) and Scorpion SUGV
- Nova 5

Autonomous UGV

An autonomous UGV is essentially an autonomous robot but is specifically a vehicle that operates on the surface of the ground.

A fully autonomous robot in the real world has the ability to:

- Gain information about the environment.
- Work for extended durations without human intervention.
- Travel from point A to point B, without human navigation assistance.
- Avoid situations that are harmful to people, property or itself, unless those are part of its design specifications
- Repair itself without outside assistance.
- Detect objects of interest such as people and vehicles.

A robot may also be able to learn autonomously. Autonomous learning includes the ability to:

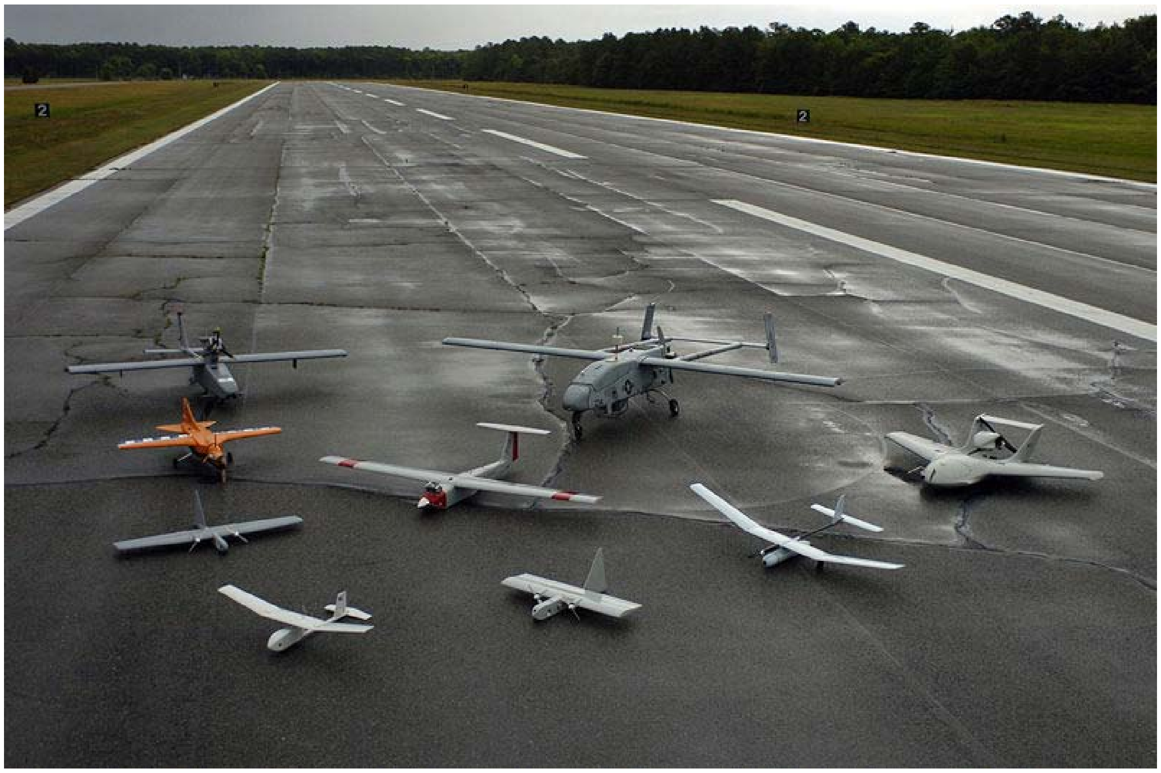
- Learn or gain new capabilities without outside assistance.
- Adjust strategies based on the surroundings.
- Adapt to surroundings without outside assistance.

Autonomous robots still require regular maintenance, as with all machines.

Aerial autonomous logistics

Based on Unmanned aerial vehicle technology, aerial autonomous logistics (or logistics UAVs) provides transfer of resources and equipment in disaster relief situations, replenishment operations, reconnaissance operations where information is gathered, and general parcel or package delivery.

Unmanned aerial vehicle



A group photo of aerial demonstrators at the 2005 Naval Unmanned Aerial Vehicle Air Demo.

An **unmanned aerial vehicle** (**UAV**; also known as a **remotely piloted vehicle** or **RPV**, or Unmanned Aircraft System (**UAS**)) is an aircraft that is flown by a pilot or a navigator (now called Combat Systems Officer) depending on the different Air Forces; however, without a human crew on board the aircraft. Their largest uses are in military applications. To distinguish UAVs from missiles, a UAV is defined as a powered, aerial vehicle that does not carry a human operator, uses aerodynamic forces to provide vehicle lift, can fly autonomously or be piloted remotely, can be expendable or recoverable, and can carry a lethal or nonlethal payload. Therefore, cruise missiles are not considered UAVs, because, like many other guided missiles, the vehicle itself is a weapon that is not reused, even though it is also unmanned and in some cases remotely guided.

There are a wide variety of UAV shapes, sizes, configurations, and characteristics. Historically, UAVs were simple **drones** (remotely piloted aircraft), but autonomous control is increasingly being employed in UAVs. UAVs come in two varieties: some are controlled from a remote location, and others fly autonomously based on pre-programmed flight plans using more complex dynamic automation systems.

Currently, military UAVs perform reconnaissance as well as attack missions. While many successful drone attacks on militants have been reported, they are also prone to collateral damage and/or erroneous targeting, as with many other weapon types. UAVs are also used in a small but growing number of civil applications, such as firefighting or nonmilitary security work, such as surveillance of pipelines. UAVs are often preferred for missions that are too "dull, dirty, or dangerous" for manned aircraft.

The abbreviation UAV has been expanded in some cases to **UAVS** (**u**nmanned-**a**ircraft **v**ehicle **s**ystem). In the United States, the Federal Aviation Administration has adopted the generic class unmanned aircraft system (**UAS**) originally introduced by the U.S. Navy to reflect the fact that these are not just aircraft, but *systems,* including ground stations and other elements.

History

Ryan Firebee was a series of target drones/unmanned aerial vehicles.

The earliest unmanned aerial vehicle was A. M. Low's "Aerial Target" of 1916. A number of remote-controlled airplane advances followed, including the Hewitt-Sperry Automatic Airplane, during and after World War I, including the first scale RPV (Remote Piloted Vehicle), developed by the film star and model airplane enthusiast Reginald Denny in 1935. More were made in the technology rush during the Second World War; these were used both to train antiaircraft gunners and to fly attack missions. Jet engines were applied after WW2, in such types as the Teledyne Ryan Firebee I of 1951, while companies like Beechcraft also got in the game with their Model 1001 for the United States Navy in 1955. Nevertheless, they were little more than remote-controlled airplanes until the Vietnam Era.

The birth of US UAVs (called RPVs at the time) began in 1959 when USAF officers, concerned about losing US pilots over hostile territory, began planning for the use of unmanned flights. This plan became intensified when Francis Gary Powers and his "secret" U-2 were shot down over the USSR in 1960. Within days, the highly classified UAV program was launched under the code name of "Red Wagon." The August 2 and August 4, 1964, clash in the Tonkin Gulf between naval units of the US and North Vietnamese Navy initiated America's *highly classified* UAVs into their first combat missions of the Vietnam War. Indeed, the USAF's UAVs were so secret, than even when the "Red Chinese" showed photographs of downed US UAVs via *Wide World Photos*, the official US response was, "no comment." Only on February 26, 1973, during testimony before the US House Appropriations Committee, did the US military *officially* confirm that they had been utilizing UAVs in Southeast Asia (Vietnam). While over 5,000 US airmen had been killed and over 1,000 more were either missing in action (MIA), or captured (prisoners of war/POW); the USAF 100th Strategic Reconnaissance Wing had flown approximately 3,435 UAV missions during the war, at a cost of about 554 UAVs lost to all causes. In the words of USAF General George S. Brown, Commander, Air Force Systems Command in 1972, "The only reason we need (UAVs) is that we don't want to needlessly expend the man in the cockpit." Later that same year, General John C. Meyer, Commander in Chief, Strategic Air Command, stated, "we let the drone do the high-risk flying...the loss rate is high, but we are willing to risk more of them...*they save lives!"*

During the 1973 Yom Kippur War, Syrian missile batteries in Lebanon caused heavy damages to Israeli fighter jets. As a result, Israel developed their first modern UAV. The images provided by these UAVs helped Israel to completely neutralize the Syrian air defenses at the start of the 1982 Lebanon War, resulting in no pilots downed.

With the maturing and miniaturization of applicable technologies as seen in the 1980s and 1990s, interest in UAVs grew within the higher echelons of the US military. UAVs were seen to offer the possibility of cheaper, more capable fighting machines that could be used without risk to aircrews. Initial generations were primarily surveillance aircraft, but some were armed (such as the MQ-1 Predator, which utilized AGM-114 Hellfire air-to-ground missiles). An armed UAV is known as an unmanned combat air vehicle (UCAV).

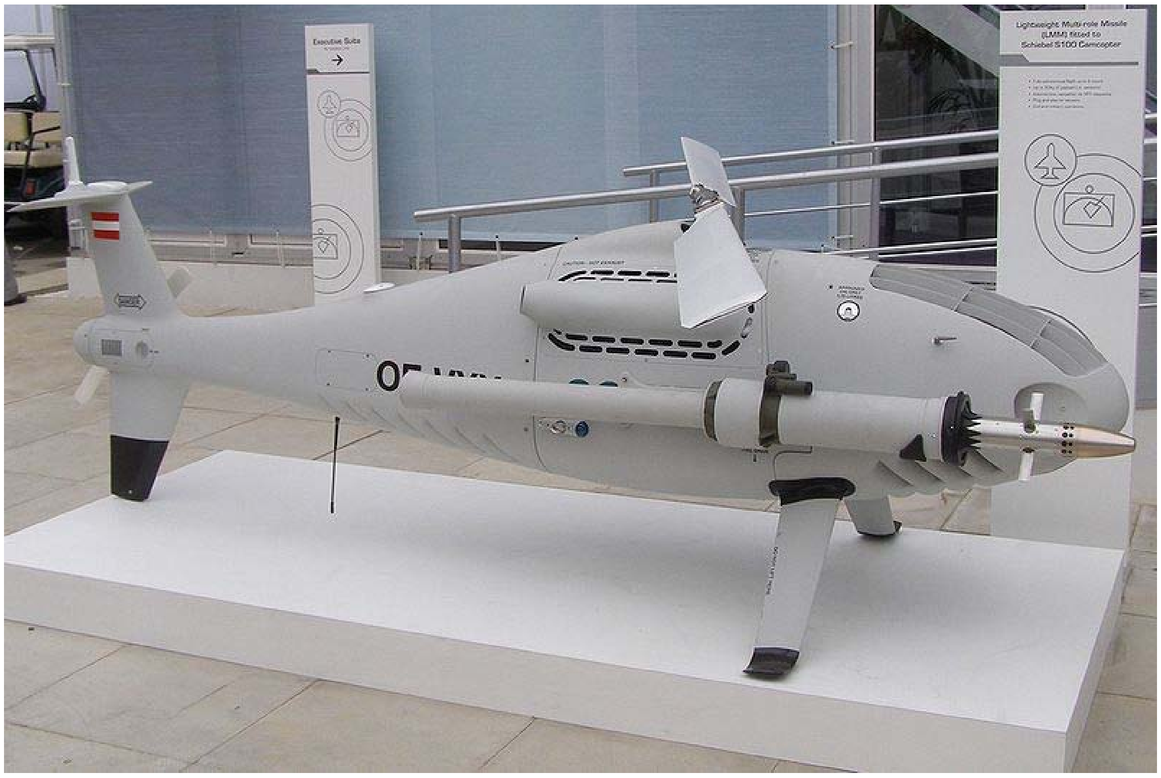
As a tool for search and rescue, UAVs can help find humans lost in the wilderness, trapped in collapsed buildings, or adrift at sea.

UAV classification

Although most UAVs are fixed-wing aircraft, rotorcraft designs (i.e., RUAVs) such as this MQ-8B Fire Scout are also used.

UAVs typically fall into one of six functional categories (although multi-role airframe platforms are becoming more prevalent):

- Target and decoy – providing ground and aerial gunnery a target that simulates an enemy aircraft or missile
- Reconnaissance – providing battlefield intelligence
- Combat – providing attack capability for high-risk missions
- Logistics – UAVs specifically designed for cargo and logistics operation
- Research and development – used to further develop UAV technologies to be integrated into field deployed UAV aircraft
- Civil and Commercial UAVs – UAVs specifically designed for civil and commercial applications


Schiebel S-100 fitted with a Lightweight Multirole Missile

They can also be categorised in terms of range/altitude and the following has been advanced as relevant at such industry events as ParcAberporth Unmanned Systems forum:

- Handheld 2,000 ft (600 m) altitude, about 2 km range
- Close 5,000 ft (1,500 m) altitude, up to 10 km range

- NATO type 10,000 ft (3,000 m) altitude, up to 50 km range
- Tactical 18,000 ft (5,500 m) altitude, about 160 km range
- MALE (medium altitude, long endurance) up to 30,000 ft (9,000 m) and range over 200 km
- HALE (high altitude, long endurance) over 30,000 ft (9,100 m) and indefinite range
- HYPERSONIC high-speed, supersonic (Mach 1–5) or hypersonic (Mach 5+) 50,000 ft (15,200 m) or suborbital altitude, range over 200 km
- ORBITAL low earth orbit (Mach 25+)
- CIS Lunar Earth-Moon transfer
- CACGS Computer Assisted Carrier Guidance System for UAVs

Additional category can be applied in pattern of function: fixed routes vs. dynamically variable routes:

- A Train Cable UAV (TCUAV) is a combination of three concepts: unmanned aerial vehicles (UAVs), unmanned ground vehicles (UGVs), and trains.

The United States military employs a tier system for categorizing its UAVs.

United States military UAV classifications

The modern concept of U.S. military UAVs is to have the various aircraft systems work together in support of personnel on the ground. The integration scheme is described in terms of a "Tier" system, and is used by military planners to designate the various individual aircraft elements in an overall usage plan for integrated operations. The Tiers do not refer to specific models of aircraft, but rather roles for which various models and their manufacturers competed. The U.S. Air Force and the U.S. Marine Corps each has its own tier system, and the two systems are themselves not integrated.

US Air Force tiers

A MQ-9 Reaper, a hunter-killer surveillance UAV.

- Tier N/A: Small/Micro UAV. Role filled by BATMAV (Wasp Block III).
- Tier I: Low altitude, long endurance. Role filled by the Gnat 750.
- Tier II: Medium altitude, long endurance (MALE). Role currently filled by the MQ-1 Predator and MQ-9 Reaper.

- Tier II+: High altitude, long endurance conventional UAV (or HALE UAV). Altitude: 60,000 to 65,000 feet (19,800 m), less than 300 knots (560 km/h) airspeed, 3,000-nautical-mile (6,000 km) radius, 24 hour time-on-station capability. Complementary to the Tier III- aircraft. Role currently filled by the RQ-4 Global Hawk.
- Tier III-: High altitude, long endurance low-observable UAV. Same parameters as, and complementary to, the Tier II+ aircraft. The RQ-3 DarkStar was originally intended to fulfill this role before it was "terminated." On December 4, 2009 the USAF confirmed the existence of the RQ-170 Sentinel.

US Marine Corps tiers

- Tier N/A: Micro UAV. Wasp III fills this role, driven largely by the desire for commonality with the USAF BATMAV.
- Tier I: Role currently filled by the Dragon Eye but all ongoing and future procurement for the Dragon Eye program is going now to the RQ-11B Raven B.
- Tier II: Role currently filled by the ScanEagle and, to some extent, the RQ-2 Pioneer.
- Tier III: For two decades, the role of medium range tactical UAV was filled by the Pioneer UAV. In July 2007, the Marine Corps announced its intention to retire the aging Pioneer fleet and transition to the Shadow Tactical Unmanned Aircraft System by AAI Corporation. The first Marine Shadow systems have already been delivered, and training for their respective Marine Corps units is underway.

U.S. Army tiers

- Tier I: Small UAV. Role filled by the RQ-11A/B Raven.
- Tier II: Short Range Tactical UAV. Role filled by the RQ-7A/B Shadow 200.
- Tier III: Medium Range Tactical UAV. Role currently filled by the RQ-5A / MQ-5A/B Hunter and IGNAT/IGNAT-ER, but transitioning to the Extended Range Multi-Purpose (ERMP) MQ-1C Warrior.

Future Combat Systems (FCS) (U.S. Army) classes

- Class I: For small units. Role to be filled by all new UAV with some similarity to Micro Air Vehicle.
- Class II: For companies (cancelled).
- Class III: For battalions (cancelled).
- Class IV: For brigades. Role to be filled by the RQ-8A/B / MQ-8B Fire Scout.

Unmanned aircraft system

UAS, or unmanned aircraft system, is the official United States Federal Aviation Administration (FAA) term for an unmanned aerial vehicle. Initially coined by the FAA in 2004 to reflect the fact that these complex systems include ground stations and other elements besides the actual aircraft, the term was first officially used by the FAA in early

2005 and subsequently adopted by DoD that same year in their Unmanned Aircraft System Roadmap 2005–2030. Many people have mistakenly used the term Unmanned *Aerial* System, or Unmanned *Air Vehicle* System, as these designations were in provisional use at one time or another. The inclusion of the term aircraft emphasizes that regardless of the location of the pilot and flightcrew, the operations must comply with the same regulations and procedures as do those aircraft with the pilot and flightcrew onboard. The official acronym 'UAS' is also used by International Civil Aviation Organization (ICAO) and other government aviation regulatory organizations.

Predator launching a Hellfire missile

The military role of unmanned aircraft systems is growing at unprecedented rates. In 2005, tactical- and theater-level unmanned aircraft alone had flown over 100,000 flight hours in support of Operation Enduring Freedom and Operation Iraqi Freedom, in which they are organized under Task Force Liberty in Afghanistan and Task Force ODIN in Iraq. Rapid advances in technology are enabling more and more capability to be placed on smaller airframes which is spurring a large increase in the number of Small Unmanned Aircraft Systems (SUAS) being deployed on the battlefield. The use of SUAS in combat is so new that no formal DoD wide reporting procedures have been established to track SUAS flight hours. As the capabilities grow for all types of UAS, nations continue to subsidize their research and development leading to further advances enabling them to perform a multitude of missions. UAS no longer only perform intelligence, surveillance, and reconnaissance missions, although this still remains their predominant type. Their roles have expanded to areas including electronic attack, strike missions, suppression and/or destruction of enemy air defense, network node or communications relay, combat search and rescue, and derivations of these themes. These UAS range in cost from a few thousand dollars to tens of millions of dollars, with aircraft ranging from less than one pound to over 40,000 pounds.

When the Obama administration announced in December 2009 the deployment of 30,000 new troops in Afghanistan, there was already an increase of attacks by pilotless Predator drones against Taliban and Al Qaeda militants in Afghanistan and Pakistan's tribal areas, of which one probably killed a key member of Al Qaeda. However, neither Osama bin Laden nor Ayman al-Zawahiri was the likely target, according to reports. According to a report of the New America Foundation, armed drone strikes had dramatically increased under President Obama – even before his deployment decision. There were 43 such attacks between January and October 2009. The report draws on what it deems to be *"credible"* local and national media stories about the attacks. That compared with a total of 34 in all of 2008, President Bush's last full year in office. Since 2006, drone-launched missiles allegedly had killed between 750 and 1,000 people in Pakistan, according to the report. Of these, about 20 people were said to be leaders of Al Qaeda, Taliban, and associated groups. Overall, about 66 to 68 percent of the people killed were militants, and between 31 and 33 percent were civilians. US officials disputed the assertion that up to 30 percent of the victims of the unmanned aerial vehicle attacks were civilians.

UAV functions

UAVs perform a wide variety of functions. The majority of these functions are some form of remote sensing; this is central to the reconnaissance role most UAVs fulfill. Less common UAV functions include interaction and transport.

Remote sensing

A Bell Eagle Eye, offered to the US Coast Guard

UAV remote sensing functions include electromagnetic spectrum sensors, biological sensors, and chemical sensors. A UAV's electromagnetic sensors typically include visual spectrum, infrared, or near infrared cameras as well as radar systems. Other electromagnetic wave detectors such as microwave and ultraviolet spectrum sensors may also be used, but are uncommon. Biological sensors are sensors capable of detecting the airborne presence of various microorganisms and other biological factors. Chemical sensors use laser spectroscopy to analyze the concentrations of each element in the air.

Transport

UAVs can transport goods using various means based on the configuration of the UAV itself. Most payloads are stored in an internal payload bay somewhere in the airframe. For many helicopter configurations, external payloads can be tethered to the bottom of the airframe. With fixed wing UAVs, payloads can also be attached to the airframe, but aerodynamics of the aircraft with the payload must be assessed. For such situations, payloads are often enclosed in aerodynamic pods for transport.

Scientific research

The RQ-7 Shadow is capable of delivering a 20 lb (9.1 kg) "Quick-MEDS" canister to front-line troops

Unmanned aircraft are uniquely capable of penetrating areas which may be too dangerous for piloted craft. The National Oceanic and Atmospheric Administration (NOAA) began utilizing the Aerosonde unmanned aircraft system in 2006 as a hurricane hunter. AAI Corporation subsidiary Aerosonde Pty Ltd. of Victoria (Australia), designs and manufactures the 35-pound system, which can fly into a hurricane and communicate near-real-time data directly to the National Hurricane Center in Florida. Beyond the standard barometric pressure and temperature data typically culled from manned hurricane hunters, the Aerosonde system provides measurements far closer to the water's surface than previously captured. Further applications for unmanned aircraft can be explored once solutions have been developed for their accommodation within national airspace, an issue currently under discussion by the Federal Aviation Administration. UAVSI, the UK manufacturer also produce a variant of their Vigilant light UAS (20 kg) designed specifically for scientific research in severe climates such as the Antarctic.

Fulmar UAV, developed by Aerovision for civilian applications

UAV Stardust II, developed under sUAS ARC FAA

Armed attacks

IAI Heron , an Unmanned Aerial Vehicle developed by the Malat (UAV) division of Israel Aerospace Industries.

MQ-1 Predator UAVs armed with Hellfire missiles are now used as platforms for hitting ground targets in sensitive areas. Armed Predators were first used in late 2001 from bases in Pakistan and Uzbekistan, mostly for hitting high profile individuals (terrorist leaders etc.') inside Afghanistan. Since then, there were several reported cases of such attacks taking place in Pakistan, this time from Afghan-based Predators. The advantage of using an unmanned vehicle, rather than a manned aircraft in such cases, is to avoid a diplomatic embarrassment should the aircraft be shot down and the pilots captured, since the bombings took place in countries deemed friendly and without the official permission of those countries.

A Predator, based in a neighboring Arab country, was used to kill suspected al-Qaeda terrorists in Yemen on November 3, 2002. This marked the first use of an armed Predator as an attack aircraft outside of a theater of war such as Afghanistan.

Questions have been raised about the accuracy of the targeting of UAVs. In March 2009, *The Guardian* reported that Israeli UAVs armed with missiles killed 48 Palestinian civilians in the Gaza Strip, including two small children in a field and a group of women and girls in an otherwise empty street. In June, Human Rights Watch investigated six UAV attacks which resulted in civilian casualties, and found that Israeli forces either failed to take all feasible precautions to verify that the targets were combatants, or failed to distinguish between combatants and civilians. In July 2009, Brookings Institution released a report stating that in the United States-led drone attacks in Pakistan, ten civilians died for every militant killed. S. Azmat Hassan, a former ambassador of Pakistan, said in July 2009 that American UAV attacks were turning Pakistani opinion against the United States, and that 35 or 40 such attacks only killed 8 or 9 top al-Qaeda operatives.

One issue with civilian casualties is the relative lack of discretion of the 100 lb (45 kg) Hellfire, which was designed to eliminate tanks and attack bunkers. The Raytheon Griffin is being fielded as a more discrete alternative, and development is underway on the still smaller, US Navy-developed Spike missile. The payload-limited Predator A can also be armed with six Griffin missiles, as opposed to only two of the much-heavier Hellfires.

Search and rescue

UAVs will likely play an increased role in search and rescue in the United States. This was demonstrated by the successful use of UAVs during the 2008 hurricanes that struck Louisiana and Texas.

For example, Predators, operating between 18,000–29,000 feet above sea level, performed search and rescue and damage assessment. Payloads carried were an optical sensor (which is a daytime and infra red camera) and a synthetic aperture radar. The Predator's SAR is a sophisticated all-weather sensor capable of providing photographic-like images through clouds, rain or fog, and in daytime or nighttime conditions; all in real-time. A concept of coherent change detection in SAR images allows for exceptional search and rescue ability: photos taken before and after the storm hits are compared and a computer highlights areas of damage.

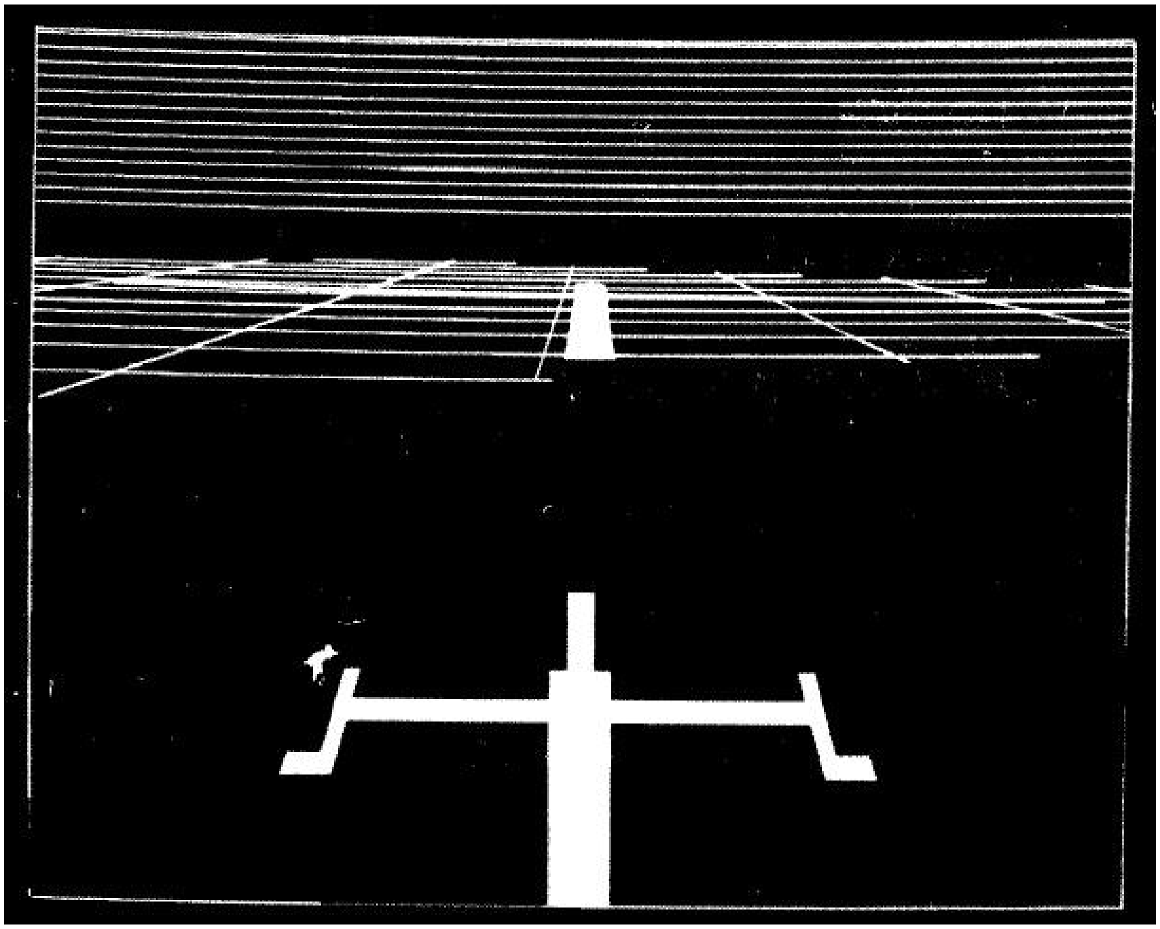
Design and development considerations

UAV design and production is a global activity, with manufacturers all across the world. The United States and Israel were initial pioneers in this technology, and U.S. manufacturers have a market share of over 60% in 2006, with U.S. market share due to increase by 5–10% through 2016. Northrop Grumman and General Atomics are the dominant manufacturers in this industry, on the strength of the Global Hawk and Predator/Mariner systems. Israeli and European manufacturers form a second tier due to lower indigenous investments, and the governments of those nations have initiatives to acquire U.S. systems due to higher levels of capability. European market share represented just 4% of global revenue in 2006.

Development costs for American military UAVs, as with most military programs, have tended to overrun their initial estimates. This is mostly due to changes in requirements during development and a failure to leverage UAV development programs over multiple armed services. This has caused United States Navy UAV programs to increase from zero to five percent in cost while United States Air Force UAV programs have increased from 60 to 284 percent.

Degree of autonomy

UAV monitoring and control at CBP



HiMAT Remote Cockpit Synthetic Vision Display (Photo: NASA 1984)

Early UAVs used during the Vietnam War after launch captured video that was recorded to film or tape on the aircraft. These aircraft often were launched and flew either in a straight line or in preset circles collecting video until they ran out of fuel and landed. After landing, the film was recovered for analysis. Because of the simple nature of these aircraft, they were often called drones. As new radio control systems became available, UAVs were often remote controlled and the term "remotely piloted vehicle" came into vogue. Today's UAVs often combine remote control and computerized automation. More sophisticated versions may have built-in control and/or guidance systems to perform low-level human pilot duties such as speed and flight-path stabilization, and simple scripted navigation functions such as waypoint following. In news and other discussions, often the term "drone" is still mistakenly used to refer to these more sophisticated aircraft.

From this perspective, most early UAVs are not autonomous at all. In fact, the field of air-vehicle autonomy is a recently emerging field, whose economics is largely driven by the military to develop battle-ready technology. Compared to the manufacturing of UAV flight hardware, the market for autonomy technology is fairly immature and undeveloped. Because of this, autonomy has been and may continue to be the bottleneck for future

UAV developments, and the overall value and rate of expansion of the future UAV market could be largely driven by advances to be made in the field of autonomy.

Autonomy technology that is important to UAV development falls under the following categories:

- Sensor fusion: Combining information from different sensors for use on board the vehicle
- Communications: Handling communication and coordination between multiple agents in the presence of incomplete and imperfect information
- Path planning: Determining an optimal path for vehicle to go while meeting certain objectives and mission constraints, such as obstacles or fuel requirements
- Trajectory Generation (sometimes called Motion planning): Determining an optimal control maneuver to take to follow a given path or to go from one location to another
- Trajectory Regulation: The specific control strategies required to constrain a vehicle within some tolerance to a trajectory
- Task Allocation and Scheduling: Determining the optimal distribution of tasks amongst a group of agents, with time and equipment constraints
- Cooperative Tactics: Formulating an optimal sequence and spatial distribution of activities between agents in order to maximize chance of success in any given mission scenario

Autonomy is commonly defined as the ability to make decisions without human intervention. To that end, the goal of autonomy is to teach machines to be "smart" and act more like humans. The keen observer may associate this with the development in the field of artificial intelligence made popular in the 1980s and 1990s such as expert systems, neural networks, machine learning, natural language processing, and vision. However, the mode of technological development in the field of autonomy has mostly followed a bottom-up approach, such as hierarchical control systems, and recent advances have been largely driven by the practitioners in the field of control science, not computer science. Similarly, autonomy has been and probably will continue to be considered an extension of the controls field.

To some extent, the ultimate goal in the development of autonomy technology is to replace the human pilot. It remains to be seen whether future developments of autonomy technology, the perception of the technology, and most importantly, the political climate surrounding the use of such technology, will limit the development and utility of autonomy for UAV applications. Also as a result of this, synthetic vision for piloting has not caught on in the UAV arena as it did with manned aircraft. NASA utilized synthetic vision for test pilots on the HiMAT program in the early 1980s, but the advent of more autonomous UAV autopilots, greatly reduced the need for this technology.

Interoperable UAV technologies became essential as systems proved their mettle in military operations, taking on tasks too challenging or dangerous for troops. NATO addressed the need for commonality through STANAG (Standardization Agreement)

4586. According to a NATO press release, the agreement began the ratification process in 1992. Its goal was to allow allied nations to easily share information obtained from unmanned aircraft through common ground control station technology. STANAG 4586 — aircraft that adhere to this protocol are equipped to translate information into standardized message formats; likewise, information received from other compliant aircraft can be transferred into vehicle-specific messaging formats for seamless interoperability. Amendments have since been made to the original agreement, based on expert feedback from the field and an industry panel known as the Custodian Support Team. Edition Two of STANAG 4586 is currently under review. There are many systems available today that are developed in accordance with STANAG 4586, including products by industry leaders such as AAI Corporation, CDL Systems, and Raytheon, all three of which are members of the Custodian Support Team for this protocol.

Endurance

RQ-4 Global Hawk, a high-altitude reconnaissance UAV capable of 36 hours continuous flight time

Because UAVs are not burdened with the physiological limitations of human pilots, they can be designed for maximized on-station times. The maximum flight duration of unmanned, aerial vehicles varies widely. Internal-combustion-engine aircraft endurance depends strongly on the percentage of fuel burned as a fraction of total weight (the Breguet endurance equation), and so is largely independent of aircraft size. Solar-electric

UAVs hold potential for unlimited flight, a concept originally championed by the AstroFlight Sunrise in 1974 and the much later Aerovironment Helios Prototype, which was destroyed in a 2003 crash.

Electric UAVs kept aloft indefinitely by laser power-beaming technology represent another proposed solution to the endurance challenge. This approach is advocated by Jordin Kare and Thomas Nugent.

One of the major problems with UAVs is no capability for inflight refueling. Currently the US Air Force is promoting research that should end in an inflight UAV refueling capability, which should be available by 2010.

One of the uses for a high endurance UAV would be to "stare" at the battlefield for a long period of time to produce a record of events that could then be played backwards to track where improvised explosive devices (IEDs) came from. Air Force Chief of Staff John P. Jumper started a program to create these persistent UAVs, but this was stopped once he was replaced.

The Defense Advanced Research Projects Agency (DARPA) is to sign a contract on building an UAV which should have an enormous endurance capability of about 5 years. The project is entitled "Vulture" and a September 15, 2010 news release indicated DARPA's Vulture Program Enters Phase II. The developers are certain neither on the design of the UAV nor on what fuel it should run to be able to stay in air without any maintenance for such a long period of time.

Price

The price of drones range depending on their classification. Some High Altitude Long Endurance, or HALE drones can cost upwards of $500 million.

Existing UAV systems

UAVs have been developed and deployed by many countries around the world.

The export of UAVs or technology capable of carrying a 500 kg payload at least 300 km is restricted in many countries by the Missile Technology Control Regime.

At the center of the American military's continued UAV research is the MQ-X, which builds upon the capabilities of the Reaper and Predator drones. As currently conceived, the MQ-X would be a stealthier and faster fighter-plane sized UAV capable of any number of missions: high-performance surveillance; attack options, including retractable cannons and bomb or missile payloads; and cargo capacity.

Chapter- 2

Autonomous Logistics Vehicles

Automated Transfer Vehicle

Automated Transfer Vehicle	
Description	
Role:	Supply the International Space Station with propellant, water, air, payload and experiments.
Crew:	Unmanned, but human-rated.
Dimensions	
Height:	10.3 m (34 ft)
Diameter:	4.5 m (15 ft)
Launch Payload:	7,667 kg (16,900 lb)
Return Payload:	None
Mass at launch:	20,750 kg

Pressurized Volume:	48 m^3
Electrical Energy	
Source:	4 solar panel wings of 4 panels each and 40Ah rechargeable batteries
Size:	total span 22,3 m
Generated Power:	3,800 W
On-board engines	
Main engine:	4 x 490N, Aerojet (General Dynamics) Model R-4D-11
Thrusters :	28 x 220N for Altitude control & braking, Astrium Lampoldshausen
Performance	
Endurance:	Docked with the ISS for six months
Apogee:	400 km
Perigee:	300 km
Inclination:	51.6 degrees
Launch	
Location:	ESA's Guiana Space Centre, Kourou in French Guiana
Site:	ELA-3
Booster:	Ariane 5

The **Automated Transfer Vehicle** or **ATV** is an expendable, unmanned resupply spacecraft developed by the European Space Agency (ESA). ATVs are designed to supply the International Space Station (ISS) with propellant, water, air, payload and experiments. In addition, ATVs can reboost the station into a higher orbit.

The first ATV, *Jules Verne*, was launched in March 2008 and ESA has already contracted suppliers to produce four more to be flown until 2015. A total of seven ATVs could eventually be launched to the International Space Station, mission managers said. The development cost of the ATV was approximately €1.35 billion, and each ATV spacecraft costs about $300 million, not including launch costs.

Design

The ATV is designed to complement the Progress spacecraft, having three times its capacity. Like the Progress, it carries both bulk liquids and relatively fragile freight which is stored in a cargo hold kept in a pressurized shirt-sleeve environment so that astronauts can have access to it without putting on a spacesuit. The ATV pressurized cargo section is based on the Italian-built Multi-Purpose Logistics Module (MPLM), which is already in service as a Shuttle-carried 'space barge' transporting equipment to and from the Station.

The ATV docking system consists of two videometers and two telegoniometers built by Sodern, a subsidiary of EADS. Additional monitoring data is supplied by a redundant Russian-made antenna built for the Ukrainian-built Kurs, an automatic docking system similar to those used on Soyuz manned ferries and on the Progress re-supply ship. Visual imagery is provided by a camera on the Zvezda module.

Also like the Progress, the ATV will additionally serve as a container for the station's waste.

Each ATV weighs 20.7 tonnes at launch and has a cargo capacity of 8 tonnes:

- 1,500 kilograms (3,300 lb) to 5,500 kilograms (12,000 lb) of dry cargo (re-supply goods, scientific payload, etc.),
- Up to 840 kilograms (1,900 lb) of water,
- Up to 100 kilograms (220 lb) of gas (nitrogen, oxygen, air), with up to two gases per flight,
- Up to 4,700 kilograms (10,000 lb) of propellant for the *re-boost* maneuver and refueling the station. The ATV propellant used for *re-boost* (monomethylhydrazine fuel and N_2O_4 oxidizer) is of a different type from the *payload* Russian refueling propellant (UDMH fuel and N_2O_4 oxidizer).

Development

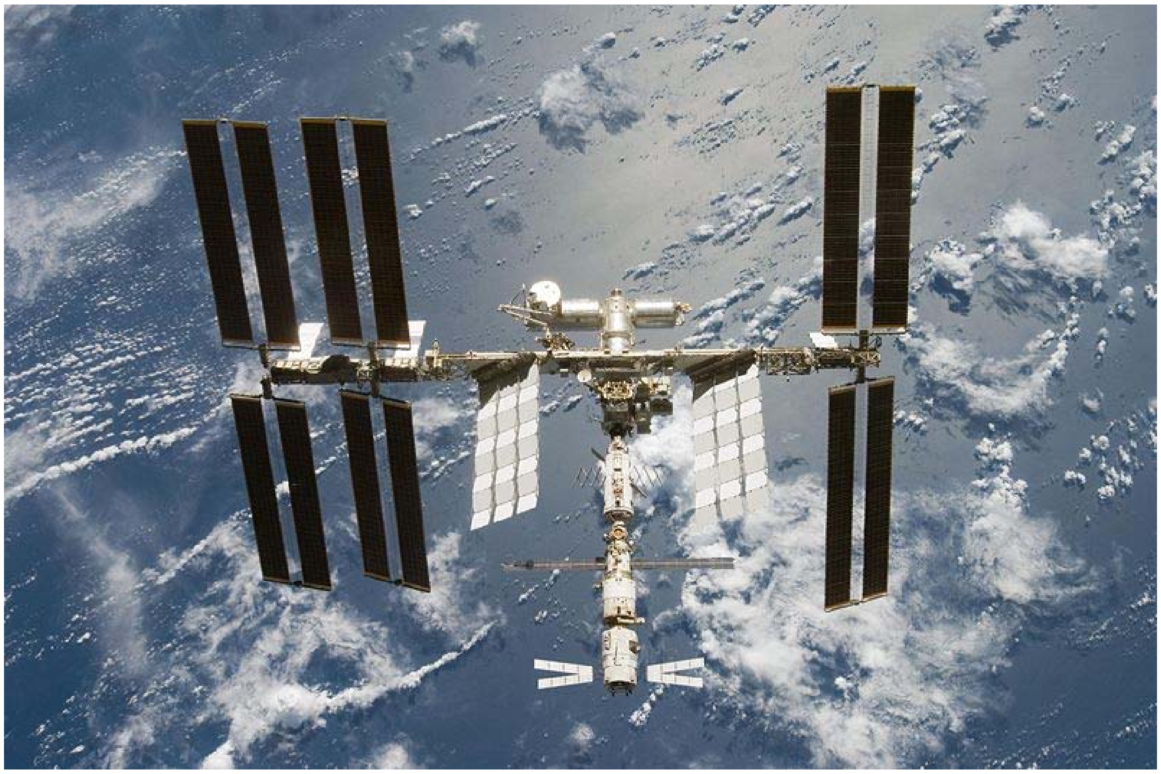
The prime contractor for the ATV is EADS Astrium Space Transportation, leading a consortium of many sub-contractors. Development was started in Les Mureaux, France and moved to Bremen, Germany, as the project moved from its development to production stage of the four initial units starts. In order to facilitate the relationship between the contractor and ESA, an integrated ESA team at the Les Mureaux site has been established for the duration of the development.

The first ATV arrived at the ESA spaceport in Kourou, French Guiana on 31 July 2007 after a nearly two week journey from Rotterdam harbour and was launched on 9 March 2008. The *Jules Verne* was the first ATV to be launched. EADS Astrium Space Transportation builds the ATVs in its facility in Bremen. Contracts and accords were signed in 2004 for four more ATVs, which should be launched about once every two years, bringing the total order, including Jules-Verne, to five.

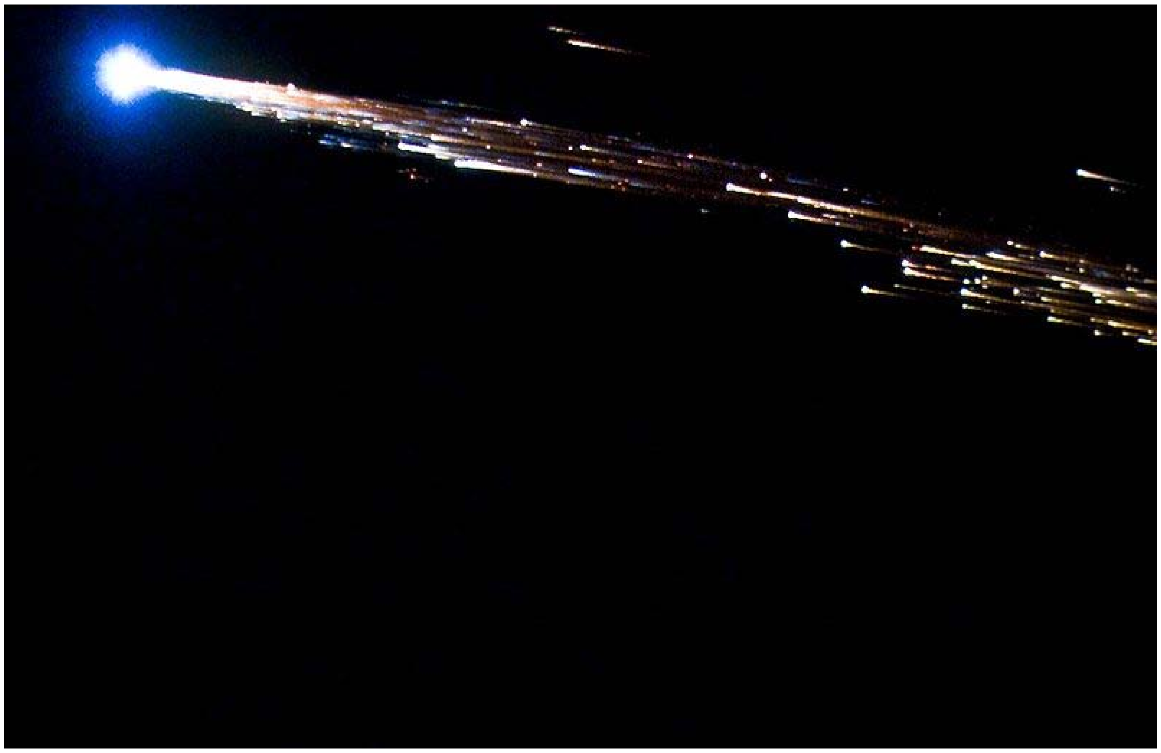
To this end, RSC Energia has signed a 40 million euro contract with one of the main subcontractors of EADS Astrium Space Transportation, the Italian company Alenia Spazio (now Thales Alenia Space), to supply the Russian Docking System, refuelling system, and Russian Equipment Control System. Within the EADS Astrium Space Transportation led project, Thales Alenia Space is in charge of the pressurized cargo carrier of the ATV. These pressurized cargo carriers are produced in Turin, Italy.

In addition to its use by ESA and Russia, the ATV was in the running to service NASA under the Commercial Orbital Transportation Services program. Under the proposal, a joint venture between EADS and Boeing, an ATV would be launched from Cape Canaveral, Florida, using a Delta IV rocket. Ultimately, it was not awarded a contract.

Use



Jules Verne seen at the bottom of the ISS making the relative size clearly visible



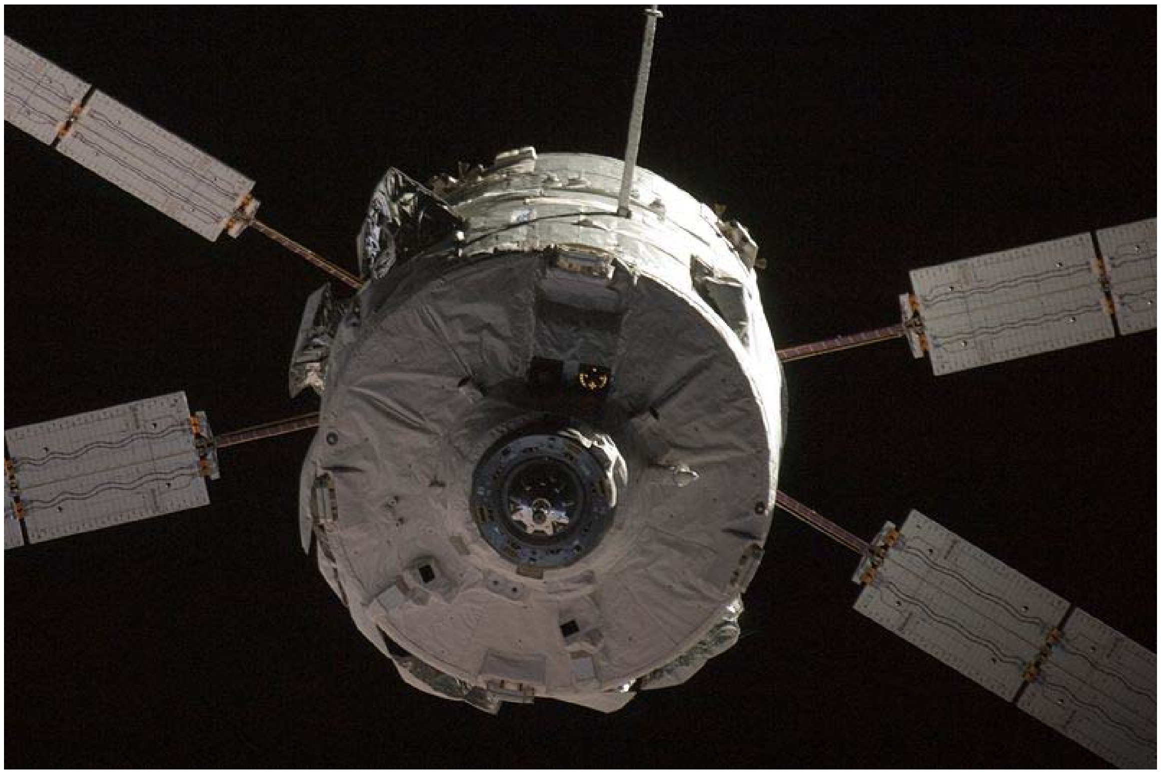
ATV *Jules Verne* as it re-enters Earth's atmosphere in a controlled burn-up after undocking from ISS.

ATVs are intended to be launched every 17 months in order to resupply the International Space Station. They use GPS and a star tracker to automatically rendezvous with the Space Station. At a distance of 249 m, the ATV computers use videometer and telegoniometer data for final approach and docking manoeuvres. The actual docking to *Zvezda* is fully automatic. If there are any last-minute problems, a pre-programmed sequence of anti-collision manoeuvres, fully independent of the main navigation system, can be activated by the flight engineers aboard the station.

With the ATV docked, the station crew enters the cargo section and removes the payload. The ATV's liquid tanks are connected to the station's plumbing and discharge their contents. The station crew manually releases air components directly into the ISS’s atmosphere. For up to six months, the ATV, mostly in dormant mode, remains attached to the ISS with the hatch remaining open. The crew then steadily fills the cargo section with the station's waste. At intervals of 10 to 45 days, the ATV’s thrusters are used to boost the station's altitude.

Once its mission is accomplished, the ATV, filled with up to 6.5 tonnes of waste, separates. Its thrusters move the spacecraft out of orbit (de-orbit) and place it on a steep flight path to perform a controlled destructive re-entry high above the Pacific Ocean.

Scheduled missions



Jules Verne Automated Transfer Vehicle approaches the International Space Station on Monday, 31 March 2008

Designation	Name	Launch date	Result	Re-entry
ATV-001	Jules Verne	9 March 2008	*Docked* 3 April 2008	29 September 2008
ATV-002	Johannes Kepler	15 February 2011	*Planned*	*Planned*
ATV-003	Edoardo Amaldi	29 February 2012	*Planned*	*Planned*
ATV-004	unnamed	February 2013	*Planned*	*Planned*
ATV-005	unnamed	February 2014	*Planned*	*Planned*

Jules Verne

The first flight of the ATV was delayed many times before its launch on 9 March 2008. It was named *Jules Verne*, in memory of the first science fiction writer of modern times. The Jules Verne carried two of the author's original handwritten manuscripts, to be received by the ISS crew as symbolic tokens of the success of the first flight.

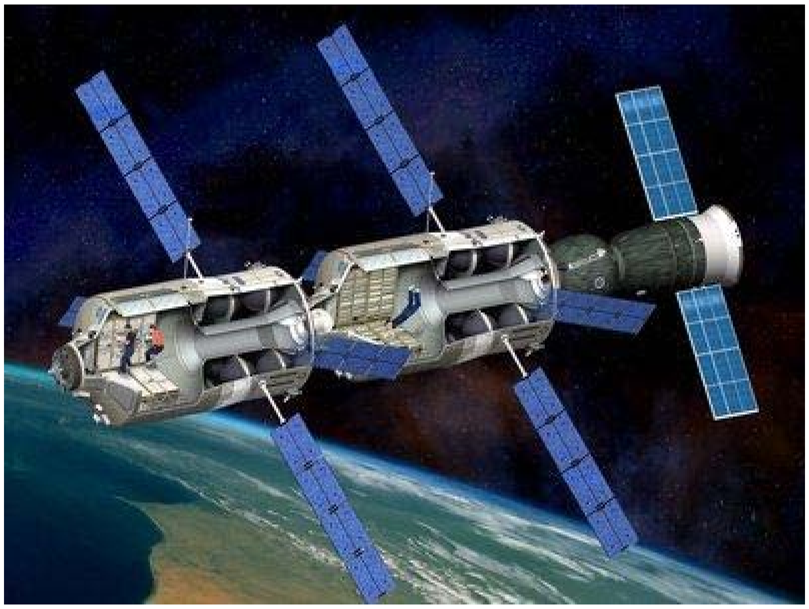
The craft was launched into a 300-kilometre (190 mi) orbit atop an Ariane 5 from the equatorial ELA-3 launch site at the Guiana Space Centre. The ATV separated from the Ariane rocket and after weeks of tests and orbit adjustments successfully docked in the International Space Station at 14:45 UTC on 3 April 2008.

In the early morning hours of 29 September 2008, the Jules Verne burnt up on entering the atmosphere above an uninhabited section of the Pacific Ocean, southwest of Tahiti.

ATV Control Centre

ATV missions are monitored and controlled from the ATV Control Centre (ATV-CC) located at the Toulouse Space Centre (CST) in Toulouse, France. The centre is responsible for all planning and executing of every orbital maneuver and mission task of the ATV, from the moment of separation from its launch vehicle, until it burns up in the Earth's atmosphere. The centre has a direct communication line with the Columbus Control Center (Col-CC) in Oberpfaffenhofen, Germany. Col-CC provides ATV-CC with access to both the American TDRSS and the European Artemis communication networks in order to communicate with ATV and the space station. ATV-CC will coordinate its actions with NASA's Mission Control Center (MCC-H) in Houston and the Russian FKA Mission Control Center (TsUP or MCC-M) in Moscow, Russia as well as the ATV launch site at the Guiana Space Centre in Kourou, French Guiana.

ATV Evolution proposals



An **MSS** could be used as a small orbital lab

Following the decision by NASA to retire the Space Shuttle around 2010, the European Space Agency launched a series of studies to determine the potential for evolutions and adaptations of the ATV. Following these studies the cargo return version (CARV) became a candidate for further development. The goal of this variant is to provide ESA with the capability to transport scientific data and cargo from the ISS to Earth. Beyond this, CARV could be enhanced to become a man-carrying spacecraft which would be launched by an adapted Ariane 5.

Mini Space Station

The MSS concept is an ATV evolution proposal for the construction of multiple ATVs with two docking ports, one at each end. The current version of the ATV is already prepared for a docking port at the back, with the main propulsion system arranged in a cylindrical fashion leaving room for a tunnel through the middle. This concept would allow Soyuz, Progress and other ATVs to dock to the back of the ATV, allowing a steady flow of Russian vehicles using the available docking ports whilst an ATV is docked for an average of around 6 months at a time.

PARES capsules would be able to hold a few kg of cargo

Payload Retrieval System

The PARES would have included a small ballistic capsule similar to VBK-Raduga embedded into the ATV docking interface, which would have brought back a few tens of kilograms of payload. PARES could have featured a deployable heat shield system. The European Space Agency was also proposing the system for use with the Progress spacecraft and the H-II Transfer Vehicle (HTV).

CARV would be used to transport a large amount of cargo to Earth.

Cargo Ascent and Return Vehicle

> The CARV study investigated a larger lifting capsule, capable of bringing back a few tonnes of payload, which could have been installed in place of the ATV pressurized cargo hold. In addition, a goal was to allow CARV to dock at the US side of the station. Given the larger docking ports there, it would be possible to transfer complete International Standard Payload Racks (ISPRs) from the ATV to the station, which is not currently possible.

Crew Transport Vehicle

> This was another option under consideration. Similar to the CARV variant, this would replace the current Integrated Cargo Carrier with a pressurized re-entry capsule. A significant difference with the cargo-only variant would be the presence of a Crew Escape System, consisting of a number of booster rockets able to pull the crew capsule away from the launcher (Ariane 5) and/or Service Module in the event of an emergency. The CTV variant of the ATV would be able to seat 4 or 5 crew.

Possibilities for launching of the ATV on other launchers than the Ariane 5 have also been investigated, in particular in the frame of Commercial Orbital Transportation Services, but NASA has meanwhile chosen a US-only solution.

Proposed crewed version

A 3D rendering of the proposed ATV derived manned transportation system.

The aerospace company EADS Astrium and the German Space Agency (the DLR), announced on 14 May 2008 that they would pursue a project to adapt the ATV into a crew transportation system. The craft would be able to launch a 3 man crew beyond LEO via use of a modified version of the Ariane 5 rocket and would be more spacious than the Russian Soyuz. A mock-up of the proposed craft was shown at the 2008 International Aerospace Exhibition in Berlin. If the project is given ESA approval development will proceed in two stages:

- The first stage would see the development of an Advanced Reentry Vehicle (ARV) capable of transporting up to 1,500 kg of cargo from space to earth safely by 2015. This capability would be available to ESA even if further development were to be halted. It would prove useful in the ISS program as well as the proposed Mars Sample Return Mission with NASA. ARV development would make use of work done on the Atmospheric Reentry Demonstrator, Crew Return Vehicle and related projects. The budget for this stage of the ATV overhaul would reportedly be €300 million.
- The second stage would adapt the then existing capsule to be able to transport people safely as well as upgrade the propulsion and other systems in the service module and would last 4 to 5 years at a cost of "a couple of billion (€)" according to a senior Astrium representative.

Subsequent activity

In November 2008, ESA ministers budgeted for a feasibility study into developing a re-entry capsule for the ATV, a requirement for developing either a cargo return capacity or a manned version of the ATV. ESA signed a €21 million study contract with EADS Astrium on 7 July 2009.

Above Water autonomous logistics

Based on Unmanned surface vehicle technology, this class of vehicles provides a range of surface fleet replenishment and equipment transfer capabilities.

Unmanned surface vehicle

The term **unmanned surface vehicle** (USV) refers to any vehicle that operates on the surface of the water without a crew. USVs have been tested since World War II but have been largely overshadowed. This is due to the fact the USVs, such as the OWL Mk II surveillance drone, have been classified as Autonomous Underwater Vehicles. Recent successes of UAVs in the Afghan War may pave the way for a new wave of USVs.

A USV demonstration at Hampton, Virginia; January 2009

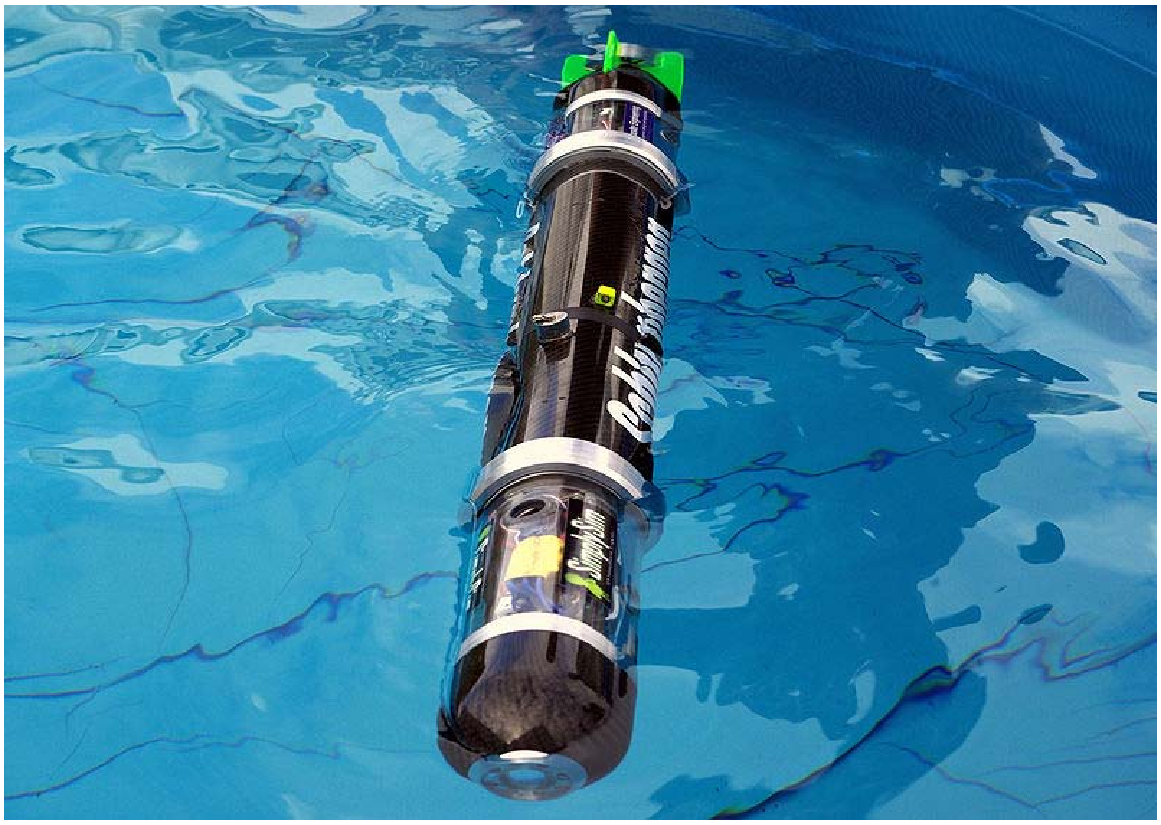
The Israeli Navy is using unmanned surface vehicles today. They are reliable, fast, highly maneuverable, allowing them to conduct a wide range of missions, including patrols of the coast, without endangering navy personnel.

Subsea autonomous logistics

Using Autonomous Underwater Vehicle technology, these vehicles provide re-supply to underwater facilities, reconnaissance of underwater structures, emergency recovery capability, and so on.

Autonomous underwater vehicle

Picture taken from the HSV Swift by an employee of Bluefin Robotics Corporation during a US Navy exercise.



The Blackghost AUV is designed to undertake an underwater assault course autonomously with no outside control.

An **autonomous underwater vehicle (AUV)** is a robot which travels underwater. In military applications, AUVs are also known as **unmanned undersea vehicles (UUVs)**. AUVs constitute part of a larger group of undersea systems known as unmanned underwater vehicles, a classification that includes non-autonomous remotely operated underwater vehicles (ROVs) – controlled and powered from the surface by an operator/pilot via an umbilical.

History

The first AUV was developed at the Applied Physics Laboratory at the University of Washington as early as 1957 by Stan Murphy, Bob Francois and later on, Terry Ewart. The "Special Purpose Underwater Research Vehicle", or SPURV, was used to study diffusion, acoustic transmission, and submarine wakes.

Other early AUVs were developed at the Massachusetts Institute of Technology in the 1970s. One of these is on display in the Hart Nautical Gallery in MIT. At the same time, AUVs were also developed in the Soviet Union (although this was not commonly known until much later).

Applications

Until relatively recently, AUVs have been used for a limited number of tasks dictated by the technology available. With the development of more advanced processing capabilities and high yield power supplies, AUVs are now being used for more and more tasks with roles and missions constantly evolving.

Commercial

The oil and gas industry uses AUVs to make detailed maps of the seafloor before they start building subsea infrastructure; pipelines and sub sea completions can be installed in the most cost effective manner with minimum disruption to the environment. The AUV allows survey companies to conduct precise surveys or areas where traditional bathymetric surveys would be less effective or too costly. Also, post-lay pipe surveys are now possible.

Military

Picture of the Pluto Plus, designed for remote minehunting and destruction by the company of Milan, Italy.

A typical military mission for an AUV is to map an area to determine if there are any mines, or to monitor a protected area (such as a harbor) for new unidentified objects. AUVs are also employed in anti-submarine warfare, to aid in the detection of manned submarines.

Research

Scientists use AUVs to study lakes, the ocean, and the ocean floor. A variety of sensors can be affixed to AUVs to measure the concentration of various elements or compounds, the absorption or reflection of light, and the presence of microscopic life.

Hobby

Many roboticists construct AUVs as a hobby. A simple AUV can be constructed for around US$99, consisting of a simple PVC pipe body and two or three waterproof motors with model airplane propellers. These AUVs can be fitted with a camera, lights and sonar like their commercial brethren, though usually their operational depth is around 50 to 100 feet, compared to the several-thousand-foot-depth capacity of some commercial models. They also tend to be less durable and are usually not oceangoing, being operated most of the time in pools or lakebeds. Several competitions exist which pit homemade AUVs against various objectives.

Vehicle designs

Bluefin-12 AUV with a Buried Object Scanning Sonar (BOSS) integrated in two wings. This picture was taken in January 2005 off the coast of Florida during engineering trials.

Hundreds of different AUVs have been designed over the past 50 or so years, but only a few companies sell vehicles in any significant numbers. There are about 10 companies that sell AUVs on the international market, including Kongsberg Maritime, Hydroid (now owned by Kongsberg), Bluefin Robotics, International Submarine Engineering Ltd. and Hafmynd.

Vehicles range in size from man portable lightweight AUVs to large diameter vehicles of over 10 metres length. Once popular amongst the military and commercial sectors, the smaller vehicles are now losing popularity. It has been widely accepted by commercial organizations that to achieve the ranges and endurances required to optimize the efficiencies of operating AUVs a larger vehicle is required. However, smaller, lightweight and less expensive AUVs are still common as a budget option for universities.

Some manufacturers have benefited from domestic government sponsorship including Bluefin and Kongsberg. The market is effectively split into three areas: scientific (including universities and research agencies), commercial offshore (oil and gas etc.) and military application (mine countermeasures, battle space preparation). The majority of these roles utilize a similar design and operate in a cruise mode. They collect data while following a preplanned route at speeds between 1 and 4 knots.

Commercially available AUVS include various designs such as the small REMUS 100 AUV developed by Woods Hole Oceanographic Institution in the US and now marketed by Hydroid, Inc.; the larger HUGIN 1000 and 3000 AUVs developed by Kongsberg Maritime and Norwegian Defence Research Establishment; the Bluefin Robotics 12-and-21-inch-diameter (300 and 530 mm) vehicles and the International Submarine Engineering Ltd. Explorer. Most AUVs follow the traditional torpedo shape as this is seen as the best compromise between size, usable volume, hydrodynamic efficiency and ease of handling. There are some vehicles that make use of a modular design, enabling components to be changed easily by the operators.

The market is evolving and designs are now following commercial requirements rather than being purely developmental. The next stage is likely to be a hybrid AUV/ROV that is capable of surveys and light intervention tasks. This requires more control and the ability to hover. Again, the market will be driven by financial requirements and the aim to save money and expensive ship time.

Today, while most AUVs are capable of unsupervised missions most operators remain within range of acoustic telemetry systems in order to maintain a close watch on their investment. This is not always possible. For example, Canada has recently taken delivery of two AUVs (ISE Explorers) to survey the sea floor underneath the Arctic ice in support of their claim under Article 76 of the United Nations Convention of the Law of the Sea. Also, ultra-low-power, long-range variants such as underwater gliders are becoming capable of operating unattended for weeks or months in littoral and open ocean areas, periodically relaying data by satellite to shore, before returning to be picked up.

As of 2008, a new class of AUVs are being developed, which mimic designs found in nature. Although most are currently in their experimental stages, these biomimetic (or bionic) vehicles are able to achieve higher degrees of efficiency in propulsion and maneuverability by copying successful designs in nature. Two such vehicles are Festo's AquaJelly and Evologics' Bionik Manta.

Sensors

Primarily oceanographic tools, AUVs carry sensors to navigate autonomously and map features of the ocean. Typical sensors include compasses, depth sensors, sidescan and other sonars, magnetometers, thermistors and conductivity probes. A demonstration at Monterey Bay in California in September 2006 showed that a 21-inch (530 mm) diameter AUV can tow a 300 feet (91 m) long hydrophone array while maintaining a 3-knot (5.6 km/h) cruising speed.12

Navigation

AUVs can navigate using an underwater acoustic positioning system. When operating within a net of sea floor deployed baseline transponders this is known as LBL navigation. When a surface reference such as a support ship is available, ultra-short baseline (USBL) or short-baseline (SBL) positioning is used to calculate where the subsea vehicle is relative to the known (GPS) position of the surface craft by means of acoustic range and bearing measurements. When it is operating completely autonomously, the AUV will surface and take its own GPS fix. Between position fixes and for precise maneuvering, an inertial navigation system on board the AUV measures the acceleration of the vehicle and Doppler velocity technology is used to measure rate of travel. A pressure sensor measures the vertical position. These observations are filtered to determine a final navigation solution. An emerging alternative is using an inertial navigation system in conjunction with either a GPS receiver, or an additional magnetic compass for Dead Reckoning whenever the GPS signal is lost.

Power

Most AUVs in use today are powered by rechargeable batteries (lithium ion, lithium polymer, nickel metal hydride etc), and are implemented with some form of Battery Management System. Some vehicles use primary batteries which provide perhaps twice the endurance—at a substantial extra cost per mission. A few of the larger vehicles are powered by aluminum based semi-fuel cells, but these require substantial maintenance, require expensive refills and produce waste product that must be handled safely. An emerging trend is to combine different battery and power systems with Ultra-capacitors.

Unmanned combat air vehicle

J-UCAS Boeing X-45A UCAV technology demonstrator

An **unmanned combat air vehicle (UCAV)** or "combat drone" is an experimental class of unmanned aerial vehicle (UAVs). They differ from ordinary UAVs, because they are designed to deliver weapons (attack targets) – possibly with a great degree of autonomy. The elimination of the need for an onboard human crew in a combat aircraft that may be shot down over enemy territory has obvious advantages for personnel safety. In addition, much equipment necessary for a human pilot (such as the cockpit, flight controls, oxygen, seat/ejection seat, etc.) can be omitted from an unmanned vehicle, resulting in a decrease in weight possibly allowing greater payloads, range and maneuverability.

Current UCAV concepts call for an aircraft which would be able to operate autonomously. It will be programmed with route and target details, and conduct the mission without help from human controllers.

Laws of war

Most countries are bound to international laws of war (such as the Geneva Conventions). These laws govern the conduct of participants in war (and also defines combatants). These laws place a burden upon participants to limit collateral damage through proper identification of targets and distinction between combatants and non-combatants. The use of completely autonomous weapon systems is problematic, however, because of the difficulty in assigning accountability to a person. Therefore, current designs still incorporate an element of human control (a "man in the loop") – meaning that a ground controller must authorize weapons release.

Concerns also include the human controller's role, because if he is a civilian and not a member of the military (which is quite possible with developmental and highly sophisticated weapons systems) he would be considered a combatant under international law which carries a distinct set of responsibilities and consequences. It is for this reason that the "man in the loop" should ideally be a member of the military that understands and accepts his role as combatant.

On 28 October 2009, United Nations Special Rapporteur on extrajudicial, summary or arbitrary executions, Philip Alston, presented a report to the Third Committee (social, humanitarian and cultural) of the General Assembly warning that the use of unmanned combat air vehicles for targeted killings will be regarded as a breach of international law unless the United States can demonstrate appropriate precautions and accountability mechanisms are in place.

Current concepts

A BAE Raven during flight testing

BAE Taranis

The *EADS Barracuda* on the Manching Air Base in Germany

- Northrop Grumman, X-47 (TD)
- Alenia Aeronautica, Sky-x (TD)
- BAE Systems Taranis stealth UCAV (TD)
- Dassault nEUROn stealth UCAV (TD)
- SAGEM Sperwer UCAV (see below)
- Denel Dynamics : UCAV-TD such as Bateleur (TD)
- EADS Germany & EADS Spain, EADS Barracuda stealth UAV/UCAV (TD)
- Elbit Systems Hermes 450 (see below)
- Israel Aircraft Industries, Eitan
- Israel Aircraft Industries, Harop
- burraq ucav, Pakistan
- MiG Skat
- AURA UAV
- Various Chinese UCAV concepts have also materialized. WZ-2000, UCAV versions of the Xianglong high altitude are long endurance UAV. Also, dedicated UCAV's Shenyang's Dark Sword (Anjian), and also revealed at Zhuhai 2008 was a model of a stealth strike UCAV with forward swept wings, filling a similar niche to US X-45 called the Warrior Eagle.
- General Atomics Avenger – long-endurance UCAV, surveillance/reconnaissance/attack, low-observables, first flight 4 April 2009.

- Turkish Aerospace Industries, Anka: Anka was publicized in 16.07.2010.

Note: Some of these are not aircraft prototypes but technology demonstrators (TD) that are not expected to enter service.

J-UCAS

- Boeing X-45 UCAV (TD)
- Northrop-Grumman X-47 Pegasus

The J-UCAS UCAV would use stealth technologies and carry precision-guided weapons such as the Joint Direct Attack Munition (JDAM) or precision miniature munitions, such as the Small-Diameter Bomb to suppress enemy air defenses.

Controllers could use real-time data sources, including satellites, to plan for and respond to changes on and around the battlefield.

USAF Hunter-Killer

- Scaled Composites Model 395
- Scaled Composites Model 396
- General Atomics MQ-9 Reaper (originally the Predator B)
- Aurora Flight Sciences/Israel Aircraft Industries Eagle/Heron 2
- Unnamed Lockheed Martin entry

The United States Air Force has shifted its UCAV program from medium-range tactical strike aircraft to long-range strategic bombers. The technology of the Long Range Strike program is based on the Lockheed Martin Polecat demonstrator.

History

This is the U.S. Air Force program for which several companies have developed vehicles.

Although the J-UCAS concept is a long way from the early idea of a "reusable cruise missile", that notion is apparently alive and well. In September 2003, an announcement was made that Lockheed Martin's famous "Skunk Works" was developing an air-launched UCAV named "the Minion". Details released describe it as having a launch weight of 3,400 kilograms (7,500 pounds) and able to carry a reconnaissance payload, a jammer system, a high-power microwave weapon, or four 100 kilogram (220 pound) GPS-guided small-diameter bombs. It could also be used as a decoy, though it would need to have radar-enhancement payload as it is described as extremely stealthy.

Range is given as up to 1,850 kilometers (1,000 nautical miles). Two would be carried into combat by a single strike fighter such as a Lockheed Martin F/A-22 Raptor, with one under each wing, and launched from standoff distances to attack heavily defended targets. In practice, two strike fighters are expected to be used, launching four Minions, with the pilot of one aircraft watching out for threats while the other directs the UCAVs over a line-of-sight communications link. After the mission, the Minions would return to base and land conventionally on retractable landing gear.

A vague picture released with the announcement showed the Minion to have a certain broad resemblance to various air-launched cruise missiles, such as the Anglo-French Matra-BAe Dynamics APACHE / Storm Shadow or the US AGM-158A Joint Air to Surface Standoff Missile (JASSM), which is also built by Lockheed Martin and may have some degree of commonality with the Minion. The picture showed the Minion to have a spikelike, square-sided fuselage, with pop-out wings and twin tailfins, with the engine inlet just forward of the tailfins and the exhaust just behind the tailfins. Both the intake and the exhaust are shielded by triangular covers.

Despite the stealthiness of the Minion, Lockheed Martin is designing it for low cost, to be substantially cheaper than the $400,000 USD JASSM. Rumors about a Skunk Works project involving a cruise-missile-like UCAV had been circulating for a year or two before the announcement. There were also very vague and unconfirmed rumors that the Minion was used in an operational evaluation during the invasion of Iraq in the spring of 2003.

There has been little or no mention of the Minion since that time. It is unclear if the program has been abandoned, or if it has just been placed under deeper secrecy. The second option seems plausible, since the current administration has been noted for being much more enthusiastic about military secrecy than previous administrations.

Somewhat more visibly, in the summer of 2004, the Air Force, in need of a less expensive short-term UCAV solution with a focus on endurance, opened up a competition for a "Hunter-Killer" UCAV. Specifications include:

- An operating altitude of 10.7 to 15.25 kilometers (35,000 to 50,000 ft).
- Endurance from 16 to 30 hours or more carrying a warload of 1,360 kilograms (3,000 pounds), in specific six 225 kilogram (500 pound) guided bombs.
- Fit of SAR/MTI or EO/IR sensors and laser target designator. Of course, the Hunter-Killer would be capable of performing surveillance or reconnaissance missions along with its active combat role.

Cost specifications were given as $10 million USD per aircraft and $30 million USD per "system", with each system including two aircraft and the necessary support gear. The Hunter-Killer program has attracted considerable interest and a number of interesting proposals.

Northrop Grumman has come up with two concepts. The first is the "Model 395", a militarized version of the Scaled Composites Proteus modified to a pure UAV configuration, with a sensor turret under the nose and a SAR-MTI pod under the forward fuselage, and carrying munitions on the centerline, for example tandem triple racks to carry six 225 kilogram (500 pound) munitions. With reduced fuel, it could even carry a single 2,270 kilogram (5,000 pound) bunker buster. At maximum takeoff weight, it would have a ceiling of 15,000 meters (49,000 ft).

The other Northrop Grumman proposal is effectively a half-weight Global Hawk, the "Model 396", with a wingspan of 10.7 meters (35 ft), a length of 27 meters (88.6 ft), and a gross weight of 6,800 kilograms (15,000 pounds), half that of the Global Hawk. It would be powered by a single Pratt & Whitney 545 bizjet turbofan.

General Atomics is of course offering the turboprop-powered Predator B for the role. Aurora Flight Sciences and Israel Aircraft Industries are offering an armed Heron 2. Lockheed Martin has responded to the Air Force request but has been keeping quiet about their proposals. Boeing did not submit a proposal, stating the company was busy with other UCAV work.

The Air Force wants to field the Hunter-Killer by 2007 and may order up to 60 machines. The program seems focused to avoid "gold plate", and most of the avionics will likely be off-the-shelf.

Sagem Sperwer

Sagem Sperwer B (not weaponized on this photo)

The Sagem Sperwer B is a long endurance tactical UAV. The Sperwer can carry two Rafael-made Spike LR missiles for 12 hours (can be extended to 20) with a range of 200 km. All ground facilities of the Sperwer SDT (used by France, Netherlands, Sweden, Greece, Canada and Denmark) are compatible with the Sperwer B.

The **Sperwer** (Pronounced *Spehr-wuhr*, German for Sparrowhawk) is a 3-meter-long unmanned aerial vehicle manufactured by the French firm SAGEM. The aircraft is piloted remotely and can cruise at altitudes of over 16,000 feet for as long as five hours. It can send back images of targets up to 150 kilometers from its operators on the ground.

Sperwer B on its launch rail

Operational history

The Sperwer is currently in service with the French Army (61e régiment d'artillerie), the Royal Netherlands Air Force, Swedish Air Force, United States Air National Guard, Hellenic Air Force (Greece) with the Netherlands in the process of removing them from front line use.

Canadian Forces employed the Sperwer in Afghanistan between 2003 and its last mission on 18 April 2009 when it was replaced with the Israeli built IAI Heron.

The Royal Danish Army also bought Sperwer, but a series of problems forced the Ministry of Defence to cancel the programme and sell the remainder to Canada. As well the Danish Army no longer operate any aircraft and there are no plans for UAVs by the Royal Danish Air Force. Canada itself removed the Sperwers from front-line use in 2009, while the Netherlands was planning to phase its Sperwer drones out of front line use in March 2009 in favor of rented UAVs from Israel's Aeronautics Defense Systems Ltd.

Elbit Hermes 450

Elbit Hermes 450 UAV

The Israeli Air Force, which operates a squadron of Hermes 450s out of Palmachim Airbase south of Tel Aviv, has adapted the Hermes 450 for use as an assault UAV, reportedly equipping it with two Hellfire missiles or, according to various sources, two Rafael-made missiles. According to Israeli, Palestinian, Lebanese and independent reports, the Israeli assault UAV has seen extensive service in the Gaza Strip and was used intensively in the Second Lebanon War. Israel has not denied this capability, but to date, its policy has been not to officially confirm it either.

BAE Taranis

Taranis is a British demonstrator programme for unmanned combat air vehicle (UCAV) technology. It is part of the UK's Strategic Unmanned Air Vehicle (Experimental) programme (SUAV[E]). BAE describes Taranis's role in this context as following: "This £124m four year programme is part of the UK Government's Strategic Unmanned Air Vehicle Experiment (SUAVE) and will result in a UCAV demonstrator with fully integrated autonomous systems and low observable features." The Taranis demonstrator will have an MTOW of about 8000 kilograms and be of a similar size to the BAE Hawk-making it one of the world's largest UAVs – that will be stealthy, fast and be able to deploy a range of munitions over a number of targets and be able to defend itself against manned and other unmanned enemy aircraft. The first steel was cut in September 2007 and ground testing will start in early 2009. The first flight of the Taranis is planned for the first quarter of 2010. The demonstrator will have two internal weapons bays. With the inclusion of "full autonomy" the intention is thus for this platform to be able to "think for itself" for a large part of the mission.

The **BAE Systems Taranis** is a British demonstrator program for Unmanned Combat Air Vehicle (UCAV) technology. It is part of the UK's Strategic Unmanned Air Vehicle (Experimental) program (SUAV[E]).

The Strategic Unmanned Air Vehicles (Experiment) IPT, or SUAV(E) IPT is responsible for directing the work required to establish the potential of Unmanned Air Vehicles.

Named after the Celtic god of thunder *Taranis*, it will "explore and demonstrate how emerging technologies and systems can deliver battle-winning capabilities for the UK armed forces."

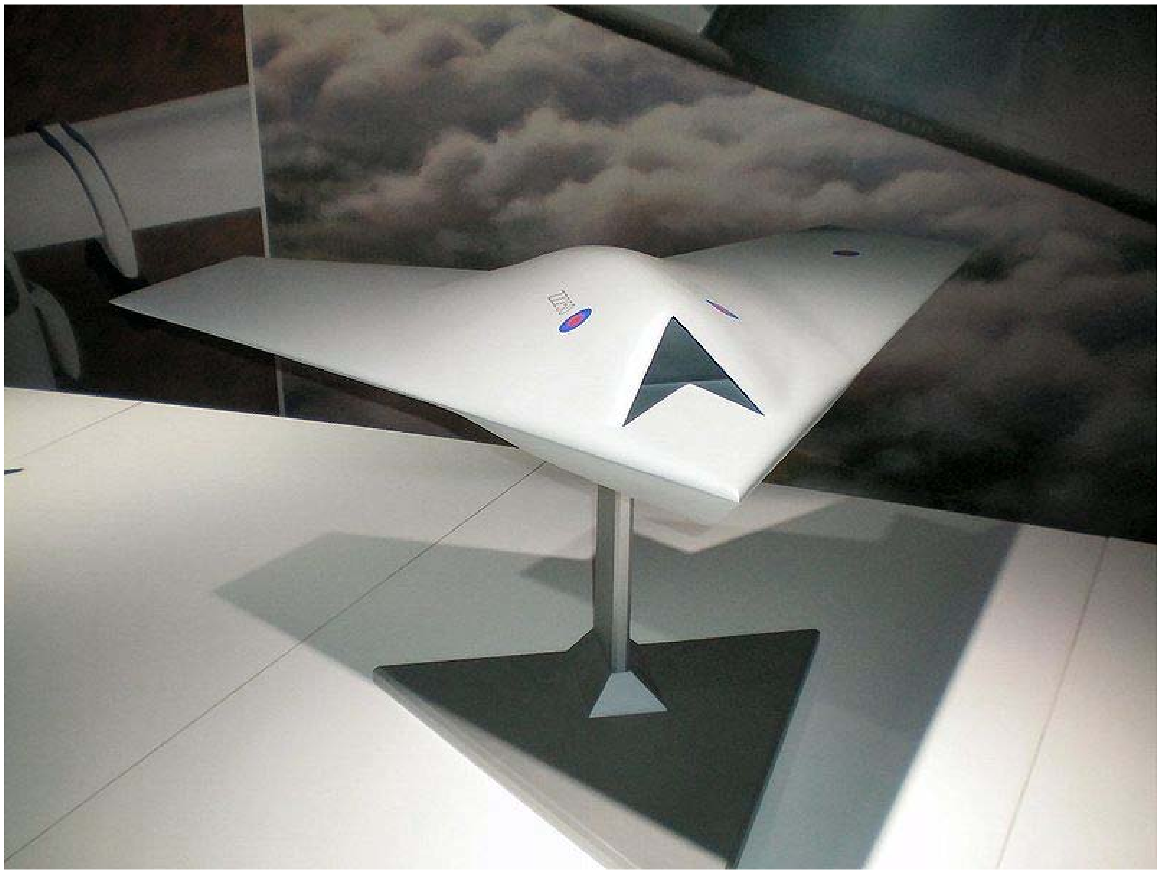
Design and development

Taranis is led by BAE Systems and also involves Rolls-Royce, GE Aviation Systems, QinetiQ and the Ministry of Defence.

As the prime contractor, BAE Systems is responsible for the overall programme and also for many of the technologies including stealth and low observability, systems integration and system control infrastructure.

BAE Systems and QinetiQ are working closely on all aspects relating to the autonomy of the system. Smiths Aerospace is responsible for providing the fuel gauging systems and the complete electrical power system for the air vehicle.

Rolls-Royce is responsible for the propulsion system and installation in the air vehicle. The aircraft is expected to use a Rolls-Royce Adour Mk.951 turbofan.



Model of Taranis on Display at Farnborough Airshow 2008 Role Autonomous UAV Manufacturer BAE Systems Primary user United Kingdom Program cost £143 million

BAE Systems Australia is tasked with developing and supplying the flight control computing.

The Integrated Systems Technologies (Insyte) subsidiary of BAE Systems, is providing C4ISTAR (computers, command, control, communications, intelligence, surveillance, target acquisition and reconnaissance) support.

BAE Systems said "Taranis will make use of at least 10 years of research and development into low observables, systems integration, control infrastructure and full autonomy. It follows the completion of risk reduction activities to ensure the mix of technologies, materials and systems used are robust enough for the 'next logical step'."

These "risk reduction activities" were related BAE programs including Replica, Nightjar I, Nightjar II, Kestrel, Corax, Raven and HERTI. BAE Systems Australia will have a workshare of about 5% in the program. The Taranis demonstrator will have a maximum takeoff weight (MTOW) of about 8,000 kilograms (18,000 lb) and be of a similar size to the BAE Hawk. The first steel was cut in September 2007 and assembly began in February 2008.

Ground testing started in early 2009.

On 9 January 2009, the MoD denied that the Taranis had been flying near the site of a smashed wind turbine, after local people claimed to have seen a UFO.

The demonstrator will have two internal weapons bays.

With the inclusion of "full autonomy" the intention is thus for this platform to be able to "think for itself" for a large part of the mission.

The development of UAVs was a key part of the UK's Defence Industrial Strategy announced in December 2005, particularly the ability of the UK to maintain its "sovereign" aircraft and UAV/UCAV skills.

Prototype unveiled

The prototype Taranis, which cost £143 million to develop, was unveiled by BAE Systems at Warton Aerodrome, Lancashire on 12 July 2010. Flight trials are expected to begin in 2011.

Specifications

Although the aircraft is still in development phase, the latest specifications which are publicly available are as follows:

- Height: 4 m
- Dimensions: 11.35 m x 9.94 m
- Weight: 8 t
- Range: Intercontinental
- Engine thrust: 6,480 lb
- Wingspan: 9.1 m

Chapter- 3

Belief-Desire-Intention Software Model

The **Belief-Desire-Intention (BDI) software model** (usually referred to simply, but ambiguously, as **BDI**) is a software model developed for programming intelligent agents. Superficially characterized by the implementation of an agent's *beliefs*, *desires* and *intentions*, it actually uses these concepts to solve a particular problem in agent programming. In essence, it provides a mechanism for separating the activity of selecting a plan (from a plan library) from the execution of currently active plans. Consequently, BDI agents are able to balance the time spent on deliberating about plans (choosing what to do) and executing those plans (doing it). A third activity, creating the plans in the first place (planning), is not within the scope of the model, and is left to the system designer and programmer.

Overview

In order to achieve this separation, the BDI software model implements the principal aspects of Michael Bratman's theory of human practical reasoning (also referred to as Belief-Desire-Intention, or BDI). That is to say, it implements the notions of belief, desire and (in particular) intention, in a manner inspired by Bratman. For Bratman, intention and desire are both pro-attitudes (mental attitudes concerned with action), but intention is distinguished as a conduct-controlling pro-attitude. He identifies commitment as the distinguishing factor between desire and intention, noting that it leads to (1) temporal persistence in plans and (2) further plans being made on the basis of those to which it is already committed. The BDI software model partially addresses these issues. Temporal persistence, in the sense of explicit reference to time, is not explored. The hierarchical nature of plans is more easily implemented: a plan consists of a number of steps, some of which may invoke other plans. The hierarchical definition of plans itself implies a kind of temporal persistence, since the overarching plan remains in effect while subsidiary plans are being executed.

An important aspect of the BDI software model (in terms of its research relevance) is the existence of logical models through which it is possible to define and reason about BDI agents. Research in this area has led, for example, to the axiomatization of some BDI implementations, as well as to formal logical descriptions such as Anand Rao and Michael Georgeff's BDICTL. The latter combines a multiple-modal logic (with modalities representing beliefs, desires and intentions) with the temporal logic CTL*. More recently, Michael Wooldridge has extended BDICTL to define LORA (the Logic Of Rational Agents), by incorporating an action logic. In principle, LORA allows reasoning not only about individual agents, but also about communication and other interaction in a multi-agent system.

The BDI software model is closely associated with intelligent agents, but does not, of itself, ensure all the characteristics associated with such agents. For example, it allows agents to have private beliefs, but does not force them to be private. It also has nothing to say about agent communication. Ultimately, the BDI software model is an attempt to solve a problem that has more to do with plans and planning (the choice and execution thereof) than it has to do with the programming of intelligent agents.

BDI Agents

A BDI agent is a particular type of bounded rational software agent, imbued with particular *mental attitudes*, viz: Beliefs, Desires and Intentions (BDI).

Architecture

This section defines the idealized architectural components of a BDI system.

- **Beliefs**: Beliefs represent the informational state of the agent, in other words its beliefs about the world (including itself and other agents). Beliefs can also include inference rules, allowing forward chaining to lead to new beliefs. Using the term *belief* rather than *knowledge* recognizes that what an agent believes may not necessarily be true (and in fact may change in the future).
    - **Beliefset**: Beliefs are stored in database (sometimes called a *belief base* or a *belief set*), although that is an implementation decision.
- **Desires**: Desires represent the motivational state of the agent. They represent objectives or situations that the agent *would like* to accomplish or bring about. Examples of desires might be: *find the best price*, *go to the party* or *become rich*.
    - **Goals**: A goal is a desire that has been adopted for active pursuit by the agent. Usage of the term *goals* adds the further restriction that the set of active desires must be consistent. For example, one should not have concurrent goals to go to a party and to stay at home - even though they could both be desirable.
- **Intentions**: Intentions represent the deliberative state of the agent - what the agent *has chosen* to do. Intentions are desires to which the agent has to some extent committed. In implemented systems, this means the agent has begun executing a plan.
    - **Plans**: Plans are sequences of actions (recipes or knowledge areas) that an agent can perform to achieve one or more of its intentions. Plans may include other plans: my plan to go for a drive may include a plan to find

    my car keys. This reflects that in Bratman's model, plans are initially only partially conceived, with details being filled in as they progress.

- **Events**: These are triggers for reactive activity by the agent. An event may update beliefs, trigger plans or modify goals. Events may be generated externally and received by sensors or integrated systems. Additionally, events may be generated internally to trigger decoupled updates or plans of activity.

BDI Interpreter

This section defines an idealized BDI interpreter that provides the basis of the PRS linage of BDI systems:

```
1. initialize-state
2. repeat
    1. options: option-generator(event-queue)
    2. selected-options: deliberate(options)
    3. update-intentions(selected-options)
    4. execute()
    5. get-new-external-events()
    6. drop-unsuccessful-attitudes()
    7. drop-impossible-attitudes()
3. end repeat
```

This basic algorithm has been extended in many ways, for instance to support planning ahead, automated teamwork, and maintenance goals.

Limitations and Criticisms

The BDI software model is one example of a reasoning architecture for a single rational agent, and one concern in a broader multi-agent system. This section bounds the scope of concerns for the BDI software model, highlighting known limitations of the architecture.

- **Learning**: BDI agents lack any specific mechanisms within the architecture to learn from past behavior and adapt to new situations.
- **Three Attitudes**: Classical decision theorists and planning researches question the necessity of having all three attitudes, distributed AI researches question whether the three attitudes are sufficient.
- **Logics**: The multi-modal logics that underlie BDI (that do not have complete axiomatizations and are not efficiently computable) have little relevance in practice.
- **Multiple Agents**: In addition to not explicitly supporting learning, the framework may not be appropriate to learning behavior. Further, the BDI model does not explicitly describe mechanisms for interaction with other agents and integration into a multi-agent system. .
- **Explicit Goals**: Most BDI implementations do not have an explicit representation of goals.

- **Lookahead**: The architecture does not have (by design) any lookahead deliberation or forward planning. This may not be desirable because adopted plans may use up limited resources, actions may not be reversible, task execution may take longer than forward planning, and actions may have undesirable side effects if unsuccessful.

BDI Agent Implementations

'Pure' BDI

Procedural Reasoning System

In Artificial Intelligence, the **Procedural Reasoning System (PRS)** is a framework for constructing real-time reasoning systems that can perform complex tasks in dynamic environments. It is based on the notion of a Rational agent or Intelligent agent using the Belief-Desire-Intention software model.

Overview

A user application is predominately defined, and provided to a PRS system is a set of *knowledge areas*. Each knowledge area is a piece of procedural knowledge that specifies how to do something, e.g., how to navigate down a corridor, or how to plan a path (in contrast with robotic architectures where the programmer just provides a model of what the states of the world are and how the agent's primitive actions affect them). Such a program, together with the PRS interpreter, is used to control the agent.

The interpreter is responsible for maintaining beliefs about the world state, choosing which goals to attempt to achieve next, and choosing which knowledge area to apply in the current situation. How exactly these operations are performed might depend on domain-specific meta-level knowledge areas. Unlike traditional AI planning systems that generate a complete plan at the beginning, and replan if unexpected things happen, PRS interleaves planning and doing actions in the world. At any point, the system might only have a partially specified plan for the future.

PRS is based on the BDI or belief-desire-intention framework for intelligent agents. Beliefs consist of what the agent believes to be true about the current state of the world, desires consist of the agent's goals, and intentions consist of the agent's current plans for achieving those goals. Furthermore, each of these three components is typically *explicitly* represented somewhere within the memory of the PRS agent at runtime, which is in contrast to purely reactive systems, such as the subsumption architecture.

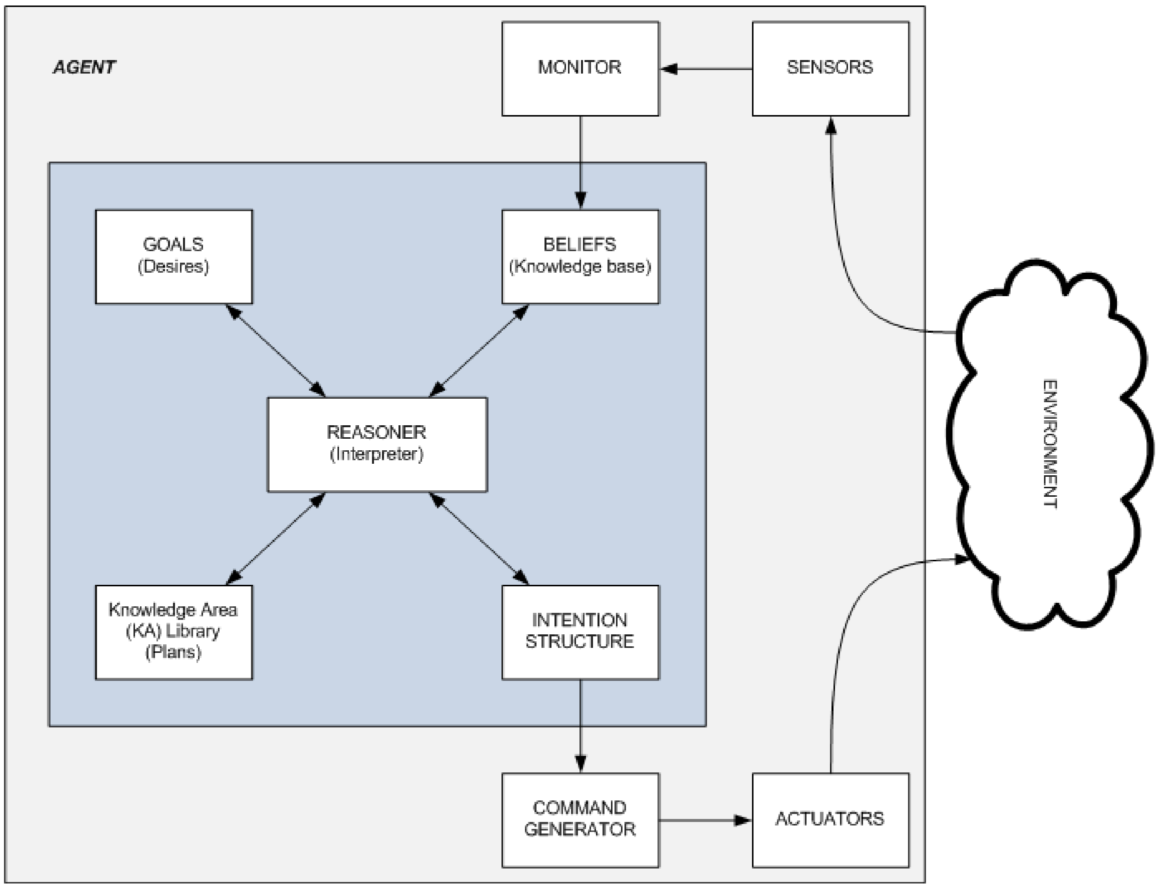
Details

This section summarizes the details of the PRS.

History

The PRS was developed by the Artificial Intelligence Center at SRI International during the 1980s. Many were involved in the development of the PRS, not limited to Michael Georgeff, Amy L. Lansky, and François Félix Ingrand. The framework was responsible for exploiting and popularizing the Belief-Desire-Intention model in software for control of an Intelligent agent. The seminal application of the framework was a fault detection system for the Reaction Control System of the NASA Space Shuttle Discovery. Development on the PRS continued at the Australian Artificial Intelligence Institute through to the late 1990s which lead to the development of a C++ implementation and extension called dMARS.

Architecture



Depiction of the PRS Architecture.

The system architecture of PRS is composed of the following components:

- **Database** for beliefs about the world, represented using first order predicate calculus.
- **Goals** to be realized by the system as conditions over an interval of time on internal and external state descriptions (beliefs).

- **Knowledge Areas** (KA) or plans that define sequences of low-level actions toward achieving a goal in specific situations.
- **Intentions** that include those KA that have been selection for current and eventual execution.
- **Interpreter** or inference mechanism that manages the system.

Features

The PRS was developed for embedded application in dynamic and real-time environments. As such it specifically addressed the limitations of other contemporary control and reasoning architectures like Expert Systems and the Blackboard system. The following define the general requirements for the development of the PRS:

- asynchronous event handling
- guaranteed reaction and response types
- procedural representation of knowledge
- handling of multiple problems
- reactive and goal-directed behavior
- focus of attention
- reflective reasoning capabilities
- continuous embedded operation
- handling of incomplete of inaccurate data
- handling of transients
- modeling delayed feedback
- operator control

Applications

The seminal application of the PRS was a monitoring and fault detection system for the Reaction Control System (RCS) on the NASA space shuttle. The RCS provides propulsive forces from a collection of jet thrusters and controls attitude of the space shuttle. A PRS-based fault diagnostic system was developed and tested using a simulator. It included over 100 KAs and over 25 meta level KAs. RCS specific KAs were written by space shuttle mission controllers. It was implemented on the Symbolics 3600 Series LISP machine and used multiple communicating instances of PRS. The system maintained over 1000 facts about the RCS, over 650 facts for the forward RCS alone and half of which are updated continuously during the mission. A version of the PRS was used to monitor the Reaction Control System on the NASA Space Shuttle Discovery.

PRS was tested on Shakey the robot including navigational and simulated jet malfunction scenarios based on the space shuttle. Later applications included a network management monitor called the Interactive Real-time Telecommunications Network Management System (IRTNMS) for Telecom Australia .

Distributed Multi-Agent Reasoning System

In artificial intelligence, the **Distributed Multi-Agent Reasoning System (dMARS)** is a platform for intelligent software agents developed at the AAII that makes uses of the BDI software model. The design for dMARS is an extension of the intelligent agent cognitive architecture developed at SRI International called PRS. The most recent incarnation of this framework is the JACK Intelligent Agents platform.

Overview

dMARS is an agent-oriented development and implementation environment written in C++ for building complex, distributed, time-critical systems.

JACK Intelligent Agents

JACK Intelligent Agents or **JACK** is a framework in Java for multi-agent system development. JACK Intelligent Agents was built by Agent Oriented Software Pty. Ltd. (AOS) and is a third generation agent platform building on the experiences of the Procedural Reasoning System (PRS) and Distributed Multi-Agent Reasoning System (dMARS). JACK is one of the few multi-agent systems that uses the BDI software model and provides its own Java-based plan language and graphical planning tools.

History

JACK Intelligent Agents was initially developed in 1997 by ex-members of the Australian Artificial Intelligence Institute (AAII or A^2I^2) who were involved in the design, implementation, and application of PRS at SRI International and/or dMARS at the AAII. The JACK platform was written for commercial application of the multi-agent paradigm (a COTS product) to complex problem solving and was the basis for starting the company Agent Oriented Software (AOS) where it remains the flagship product.

Features

JACK Intelligent Agents is a mature commercial multi-agent platform that has been under active research, development, and domain-specific application for more than 10 years. The following provides a listing of the platforms key differentiating features.

- **Agent Run-time**: The core of the platform is an extensible multi-agent run-time. Once domain specific agents, plans, events, capabilities, etc. are specified the JACK kernel manages the execution the system including message passing, reasoning, and meta-reasoning.

- **JACK Plan Language (JPL)**: JACK provides an agent-specific plan language for writing JACK plans (the discrete reasoning executed by agents). The plan language is an extension to the Java and offers commands such as @send and @post for inter-agent messaging, as well the management of actions, sub-tasks and maintenance of conditions. Plans are compiled into Java classes for execution in the JACK run-time offering speed and correctness of execution.
- **Belief-Desire-Intention Model**: In addition to a classical (non-BDI) agent model, the platform realizes the BDI software model, where beliefs are managed by beliefsets encapsulated within agents, desires are the goal states an agent is aspiring to achieve, and intentions are the meta-reasoning and plan-based reasoning the JACK agents use to achieve the current goal.
- **Capabilities**: The platform provides capabilities which are abstractions of common behaviors manifest as a complex of plans and events. Capabilities provide a way of conceptually bundling common behaviors and actions and re-using them between agents.
- **JACK Development Environment (JDE)**: Multi-agent systems can be written in Java code and the JACK plan language in a standard IDE, although the platform provides an agent-centric IDE called the JACK Development Environment or JDE. The JDE provides graphical tools for writing plans, connecting plans to agents, managing inter-agent communication, as well as compiling and running. The JDE also provides graphical tools for debugging and tracing the execution of plans and inter-agent message passing .
- **Graphical Plans**: A key feature of the JDE is the facility to write and manage Graphical Plans. These are the discrete reasoning performed by an agent represented graphically as a flow chart, allowing a programmer to manage the code performed in each step of the reasoning graph, and the subject matter expert to manage the logical flow of the reasoning based on the human-readable documentation on each node .
- **JACK Object Modeller (JACOB)**: An object serialization technology used by the JACK run-time for object initialization and inter-process communication. Java objects are serialized to human-readable ASCII text, not too dissimilar to YAML and XML .
- **Platform Independence**: The JACK platform is written in Java, allowing the deployment of JACK multi-agent systems onto the wide array of platforms that support the Java Virtual Machine.

Extensions

The JACK platform has been extended a number of times since its inception. Most of the extensions, such as JACK Teams and CoJACK were developed by or in collaboration with AOS.

- **JACK Teams**: An extension to the BDI software model that facilitates agents collaborating together in teams toward achieving a goal . Like JACK, JACK Teams supports its own plan language in what AOS refers to as *Team-oriented* *programming*. JACK Teams is integrated into and available as a part of the JACK Intelligent Agents platform .

- **CoJACK**: An extension to the JACK platform that adds a cognitive architecture to the agents for eliciting more realistic (human-like) behaviors in virtual environments .
- **FIPA JACK**: An extension to the JACK platform to support the FIPA Agent Communications Language .
- **Prometheus**: An agent-centric software engineering methodology for managing the SDLC of a multi-agent based system. JACK was used as the basis for investigation, comparison, and testing the methodology. The Prometheus involved the development of the Prometheus Design Tool (PDT) which was a GUI based tool for managing design concerns in the process .
- **JACK Eclipse Plug-in**: A plug-in that facilitates the development of JACK-based systems in the Eclipse IDE. Specifically, the plug-in adds capabilities to Eclipse to support the JACK file types (such as .plan, .agent, etc) as well support for JACK plan language.
- **JACK WebBot**: An extension that embeds the JACK kernel in the Apache Tomcat web server allowing intelligent agents to be interacted with and formulate responses HTTP requests (via the Java Servlet API) .

3APL

An Abstract Agent Programming Language or **Artificial Autonomous Agents Programming Language** or **3APL** (pronounced triple-A-P-L) is an experimental tool and programming language for the development, implementation and testing of multiple cognitive agents using the Belief-Desire-Intention (BDI) approach.

Overview

3APL was developed and is maintained by a team at the computer science department of the University of Utrecht in the Netherlands. It facilitates specification of cognitive agent behavior using actions, beliefs, goals, plans, and rules. It has been subject to at least 15 papers and conferences, and at least 4 theses.

Platform

The 3APL platform has a visual interface for the monitoring and debugging of agents being run therein, and a syntax-coloring editor for source code editing. It has been released as a Java-based software, which comes with some specification Java interfaces that can be used to develop Java-based plug-ins and libraries. These can be used to provide a visible representation of a virtual environment, for instance. A 3APL platform can also connect in client or server roles to other 3APL platforms across a network, to allow communication among 3APL agents on each platform. A lightweight version of 3APL for mobile applications, named 3APL-M "Toymaker", has also been released.

Language

The 3APL language is relatively simple. The syntax has basic boolean logical operators AND, OR and NOT, with IF-THEN-ELSE conditional statements, and WHILE-DO control flow loop structures. While temporary variables cannot be created except by calling plug-in methods or belief/goal conditions, iterative counter loops can be constructed using a combination of WHILE-DO loops, beliefs and capabilities.

A 3APL agent contains formal definitions of agent beliefs, capabilities, goals and plans. Specifically, there are six skeletal blocks that must be defined.

```
PROGRAM "agent"
BELIEFBASE {}
CAPABILITIES {}
GOALBASE {}
PLANBASE {}
PG-RULES {}
PR-RULES {}
```

The beliefs, defined using Prolog syntax, are used to remember information and to perform logical computations. Beliefs can be read by one another, edited by the capabilities, and read by conditional statements in the plans. The initial beliefs of an agent can be defined in its belief base.

```
BELIEFBASE {
        status(standby).
        at(0,0).
        location(r1,2,4).
        location(r5,6,1).
        dirty(r1).
        dirty(r5).
}
```

Capabilities define the prerequisites and effects of actions in a STRIPS-like format, reading preexisting beliefs, removing some using the NOT operator, and adding new ones by stating them.

```
CAPABILITIES {
        {status(S1)} SetStatus(S2) {NOT status(S1), status(S2)},
        {at(X1,Y1)} NowAt(X2,Y2) {NOT at(X1,Y1), at(X2,Y2)},
        {dirty(R)} Clean(R) {NOT dirty(R)}
}
```

Goals are also defined using Prolog syntax, and new goals can be adopted during runtime. Initial goals are defined in the goal base.

```
GOALBASE {
        cleanRoom(r1).
        cleanRoom(r5).
}
```

Each goal ideally has associated goal planning rules, its PG rules, which serve as an abstract plans and are called from the goals as long as their guard conditions are met.

```
PG-RULES {
        cleanRoom(R) <- dirty(R) | {
                SetStatus(cleaning(R));
                goTo(R);
                clean(R);
                SetStatus(standby);
        }
}
```

The PG rules in turn can call plan revision rules, or PR rules, which serve as subroutines, and can be called upon to execute lower level and/or repetitive tasks as long as their guard conditions are met. Initial plans are defined in the plan base, executed at the beginning of the deliberation cycle.

```
PLANBASE { SetStatus(started); }
PR-RULES {
        goTo(R) <- location(R,X,Y) AND NOT at(X,Y) | {
                NowAt(X,Y);
        }
        clean(R) <- location(R,X,Y) AND at(X,Y) | {
                Clean(R);
        }
}
```

External methods may be called to access the environments modeled in the plug-ins. However, parameters cannot be directly passed to the methods, which means that the known environment must be correspondingly modeled in the agent's beliefs. The call returns a Prolog list, which can then be processed by the agent's own predicate logic.

```
Java("JanitorWorld", moveNorth(), M);
```

Agents can also communicate with one another using *Send* commands. When a piece of information X is sent with the performative P from agent A to agent B, the sending action is recorded in A's belief base as *sent(B,P,X)* and is registered in B's belief base as *received(A,P,X)*.

```
Send(Partner,inform,dirty(R));
```

Download

3APL is available for download at the University of Utrecht's 3APL website, packaged with sample lone and communicative agents, and a discrete multi-agent foreground environment plug-in called BlockWorld.

GOAL Agent Programming Language

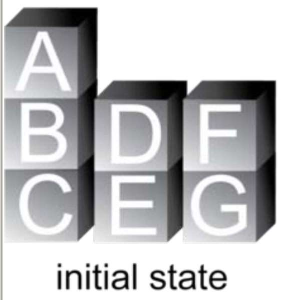
GOAL is an agent programming language for programming rational agents. GOAL agents derive their choice of action from their beliefs and goals. The language provides the basic building blocks to design and implement rational agents by means of a set of programming constructs. These programming constructs allow and facilitate the manipulation of an agent's beliefs and goals and to structure its decision-making. The language provides an intuitive programming framework based on common sense or practical reasoning.

Overview

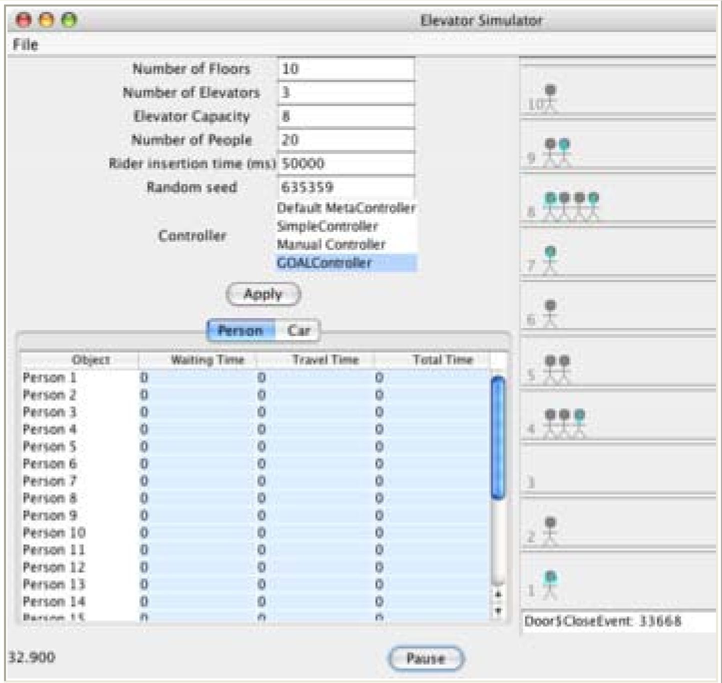
The main features of GOAL include:

- **Declarative beliefs**: Agents use a symbolic, logical language to represent the information they have, and their beliefs or knowledge about the environment they act upon in order to achieve their goals. This *knowledge representation language* is not fixed by GOAL but, in principle, may be varied according to the needs of the programmer.
- **Declarative goals**: Agents may have multiple goals that specify *what* the agent wants to achieve at some moment in the near or distant future. Declarative goals specify a state of the environment that the agent wants to establish, they do not specify actions\index{action} or procedures how to achieve such states.
- **Blind commitment strategy**: Agents commit to their goals and drop goals only when they have been achieved. This commitment strategy, called a *blind* commitment strategy in the literature, is the *default* strategy used by GOAL agents. Rational agents are assumed to not have goals that they believe are already achieved, a constraint which has been built into GOAL agents by dropping a goal when it has been *completely* achieved.
- **Rule-based action selection**: Agents use so-called *action rules* to select actions, given their beliefs and goals. Such rules may *underspecify* the choice of action in the sense that multiple actions may be performed at any time given the action rules of the agent. In that case, a GOAL agent will select an arbitrary enabled action for execution.
- **Policy-based intention modules**: Agents may focus their attention and put all their efforts on achieving a subset of their goals, using a subset of their actions, using only knowledge relevant to achieving those goals. GOAL provides modules to structure action rules and knowledge dedicated to achieving specific goals. Informally, modules can be viewed as policy-based intentions in the sense of Michael Bratman.
- **Communication at the knowledge level**: Agents may communicate with each other to exchange information, and to coordinate their actions. GOAL agents communicate using the knowledge representation language that is also used to represent their beliefs and goals.

GOAL Agent Program



An Example Blocks World Problem



Number of Floors: 10
Number of Elevators: 3
Elevator Capacity: 8
Number of People: 20
Rider insertion time (ms): 50000
Random seed: 635359
Controller: Default MetaController / SimpleController / Manual Controller / GOALController

Object	Waiting Time	Travel Time	Total Time
Person 1	0	0	0
Person 2	0	0	0
Person 3	0	0	0
Person 4	0	0	0
Person 5	0	0	0
Person 6	0	0	0
Person 7	0	0	0
Person 8	0	0	0
Person 9	0	0	0
Person 10	0	0	0
Person 11	0	0	0
Person 12	0	0	0
Person 13	0	0	0
Person 14	0	0	0
Person 15	0	0	0

Door5CloseEvent: 33668

32.900

Another Example: A GOAL Multi-Agent Elevator Controller

A GOAL agent program consists of six different sections, including the *knowledge*, *beliefs*, *goals*, *action rules*, *action specifications*, and *percept rules*, respectively. The knowledge, beliefs and goals are represented in a knowledge representation language such as Prolog, Answer set programming, SQL (or Datalog), or the Planning Domain Definition Language, for example. Below, we illustrate the components of a GOAL agent program using Prolog.

The overall structure of a GOAL agent program looks like:

```
main: <agentname> {
  <sections>
}
```

The GOAL agent code used to illustrate the structure of a GOAL agent is an agent that is able to solve Blocks world problems. The beliefs of the agent represent the current state of the Blocks world whereas the goals of the agent represent the goal state. The *knowledge* section listed next contains additional conceptual or domain knowledge related to the Blocks world domain.

```
knowledge{
  block(a), block(b), block(c), block(d), block(e), block(f), block(g).
  clear(table).
  clear(X) :- block(X), not(on(Y,X)).
  tower([X]) :- on(X,table).
  tower([X,Y|T]) :- on(X,Y), tower([Y|T]).
}
```

Note that all the blocks listed in the knowledge section reappear in the *beliefs* section again as the position of each block needs to be specified to characterize the complete configuration of blocks.

```
beliefs{
  on(a,b), on(b,c), on(c,table), on(d,e), on(e,table), on(f,g),
on(g,table).
}
```

All known blocks also are present in the *goals* section which specifies a goal configuration which reuses all blocks.

```
goals{
  on(a,e), on(b,table), on(c,table), on(d,c), on(e,b), on(f,d),
on(g,table).
}
```

A GOAL agent may have multiple goals at the same time. These goals may even be conflicting as each of the goals may be realized at different times. For example, an agent might have a goal to watch a movie in the movie theater and to be at home (afterwards).

In GOAL, different notions of goal are distinguished. A **primitive goal** is a statement that follows from the goal base in conjunction with the concepts defined in the knowledge base. For example, `tower([a,e,b])` is a primitive goal and we write `goal(tower([a,e,b])` to denote this. Initially, `tower([a,e,b])` is also an **achievement goal** since the agent does not believe that a is on top of e, e is on top of b, and b is on the table. Achievement goals are primitive goals that the agent does not believe to be the case and are denoted by `a-goal(tower([a,e,b])`. It is also useful to be able to express that a **goal has been achieved**. `goal-a(tower([e,b])` is used to express, for example, that the tower `[e,b]` has been achieved with block e on top of block b. Both achievement goals as well as the notion of a goal achieved can be defined:

```
a-goal(formula) ::= goal(formula), not(bel(formula))
goal-a(formula) ::= goal(formula), bel(formula)
```

There is a significant literature on defining the concept of an achievement goal in the agent literature.

The *program* section of a GOAL agent specifies a strategy for selecting actions by means of action rules. The first rule below states that moving block X on top of block Y (or, possibly, the table) is an option if such a move is constructive, i.e. moves the block in position. The second rule states that moving a block X to the table is an option if block X is misplaced.

```
program{
  if a-goal(tower([X,Y|T])), bel(tower([Y|T])) then move(X,Y).
  if a-goal(tower([X|T])) then move(X,table).
}
```

Actions, such as the move action used above, are specified using a STRIPS-style specification of preconditions and postconditions. A precondition specifies when the action can be performed (is enabled). A postcondition specifies what the effects of performing the action are.

```
actionspec{
  move(X,Y) {
    pre{ clear(X), clear(Y), on(X,Z), not(X=Y) }
    post{ not(on(X,Z)), on(X,Y) }
}
```

Finally, the *perceptrules* section specifies rules for processing percepts received from the environment. The rule below specifies that if a percept is received that block X is on block Y, and X is believed to be on top of Z unequal to Y, the new fact on(X,Y) is to be added to the belief base and the atom on(X,Z) is to be removed.

```
perceptrules{
  when input{ percept(on(X,Y)), on(X,Z), not(Y=Z) } do insert(on(X,Y),
not(on(X,Z))).
}
```

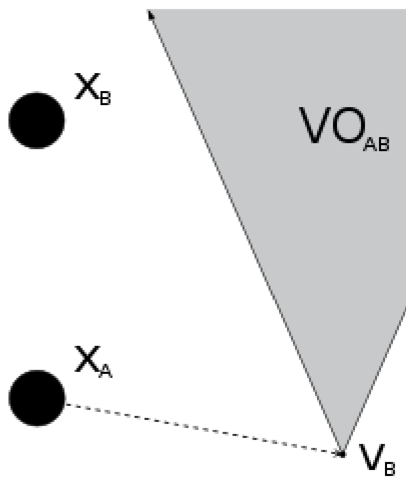
Download

GOAL is available for download from the GOAL webpage hosted at the Delft University of Technology. Besides the GOAL installer the GOAL webpage provides the GOAL Programming Guide, GOAL IDE User Manual, and a Video Tutorial.

Chapter- 4

Multi Robot Systems & Multi Agent Systems

Velocity obstacle



The velocity obstacle VO_{AB} for a robot A, with position $\mathbf{x}_A$, induced by another robot B, with position $\mathbf{x}_B$ and velocity $\mathbf{v}_B$.

In robotics and motion planning, a **velocity obstacle**, commonly abbreviated **VO**, is the set of all velocities of a robot that will result in a collision with another robot at some moment in time, assuming that the other robot maintains its current velocity. If the robot chooses a velocity inside the velocity obstacle then the two robots will eventually collide, if it chooses a velocity outside the velocity obstacle, such a collision is guaranteed not to occur.

The velocity obstacle for a robot A induced by a robot B may be formally written as

$$VO_{AB} = \{\mathbf{v} \,|\, \exists t > 0 : (\mathbf{v} - \mathbf{v}_B)t \in D(\mathbf{x}_B - \mathbf{x}_A, r_A + r_B)\}$$

where *A* has position $\mathbf{x}_A$and radius r_A, and *B* has position $\mathbf{x}_B$, radius r_B, and velocity $\mathbf{v}_B$. The notation $D(\mathbf{x}, r)$represents a disc with center $\mathbf{x}$and radius *r*.

Variations include common velocity obstacles (CVO),, finite-time-interval velocity obstacles (FVO), generalized velocity obstacles (GVO), hybrid reciprocal velocity obstacles (HRVO), nonlinear velocity obstacles (NLVO), reciprocal velocity obstacles (RVO), and recursive probabilistic velocity obstacles (PVO).

Symbrion

Symbrion (Symbiotic Evolutionary Robot Organisms) is a project funded by European Commissions to develop a framework in which a homogeneous swarm of miniature interdependent robots can co-assemble into a larger robotic organism to gain problem-solving momentum.

One of the key-aspects of Symbrion is inspired by the biological world: an artificial genome that allows to store and evolve (sub)optimal configurations in order to achieve an increased speed of adaptation.

The SYMBRION project does not start from zero, previous development and research from project I-SWARM and the open-source SWARMROBOT projects serve as a mounting point. A large part of the developments within Symbrion is open-source and open-hardware.

Co-operating universities

- Universität Stuttgart, Germany (Coordination)
- Universität Graz, Austria
- Vrije Universiteit, Netherlands
- Universität Karlsruhe, Germany
- Flanders Institute for Biotechnology, Belgium
- University of the West of England, Bristol, UK
- Eberhard Karls Universität Tübingen, Germany
- University of York, UK
- Université Libre de Bruxelles, Belgium
- Institut National de Recherche en Informatique et Automatique, France

Swarm robotics

Swarm of open-source Jasmine micro-robots recharging themselves

team of iRobot Create robots at the Georgia Institute of Technology

Swarm robotics is a new approach to the coordination of multirobot systems which consist of large numbers of mostly simple physical robots. It is supposed that a desired collective behavior emerges from the interactions between the robots and interactions of robots with the environment. This approach emerged on the field of artificial swarm intelligence, as well as the biological studies of insects, ants and other fields in nature, where swarm behaviour occurs.

Definition

The research of swarm robotics is to study the design of robots, their physical body and their controlling behaviors. It is inspired but not limited by the emergent behavior observed in social insects, called swarm intelligence. Relatively simple individual rules can produce a large set of complex swarm behavior. A key-component is the communication between the members of the group that build a system of constant feedback. The swarm behavior involves constant change of individuals in cooperation with others, as well as the behavior of the whole group.

Unlike distributed robotic systems in general, swarm robotics emphasizes a *large* number of robots, and promotes scalability, for instance by using only local communication. That local communication for example can be achieved by wireless transmission systems, like radio frequency or infrared.

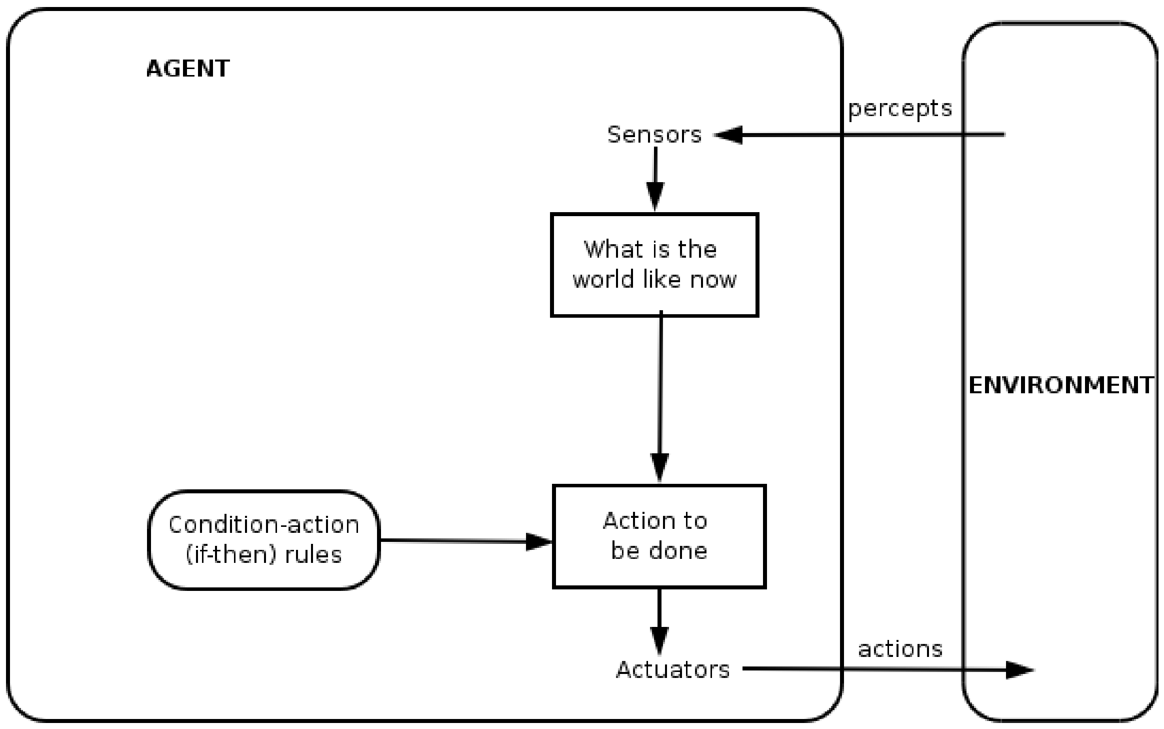
Video tracking is an essential tool for systematically studying swarm-behavior, even though other tracking methods are available. Recently Bristol robotics laboratory developed an ultrasonic position tracking system for swarm research purposes. Further research is needed to find methodologies that allow the design and reliable prediction of swarm behavior when only the features of the individual swarm members are given.

Goals and applications

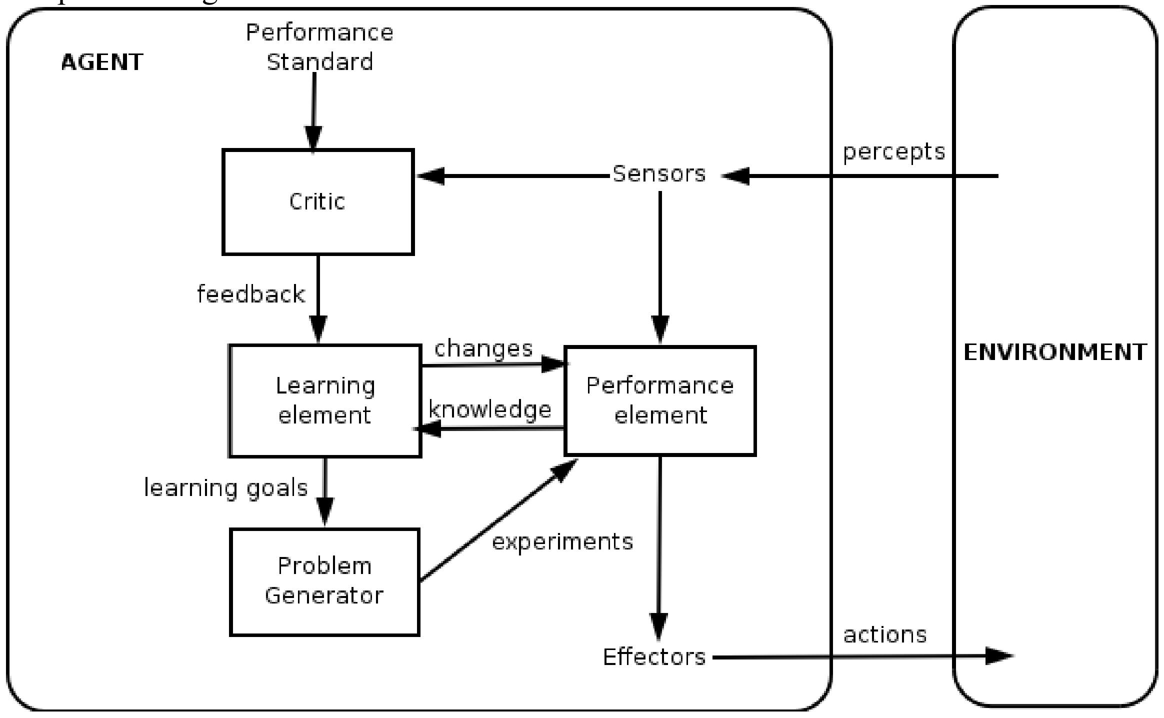
Both miniaturization and cost are key-factors in swarm robotics. These are the constraints in building large groups of robotics; therefore the simplicity of the individual team member should be emphasized. This should motivate a swarm-intelligent approach to achieve meaningful behavior at swarm-level, instead of the individual level.

Potential applications for swarm robotics include tasks that demand for miniaturization (nanorobotics, microbotics), like distributed sensing tasks in micromachinery or the human body. On the other hand swarm robotics can be suited to tasks that demand cheap designs, for instance mining tasks or agricultural foraging tasks. Also some artists use swarm robotic techniques to realize new forms of interactive art.

Multi-agent system



Simple reflex agent



Learning agent

A **multi-agent system** (**MAS**) is a system composed of multiple interacting intelligent agents. Multi-agent systems can be used to solve problems which are difficult or impossible for an individual agent or monolithic system to solve. Examples of problems which are appropriate to multi-agent systems research include online trading, disaster response, and modelling social structures.

Overview

The agents in a multi-agent system have several important characteristics:

- **Autonomy**: the agents are at least partially autonomous
- **Local views**: no agent has a full global view of the system, or the system is too complex for an agent to make practical use of such knowledge
- **Decentralization**: there is no designated controlling agent (or the system is effectively reduced to a monolithic system)

Typically multi-agent systems research refers to software agents. However, the agents in a multi-agent system could equally well be robots, humans or human teams. A multi-agent system may contain combined human-agent teams.

Multi-agent systems can manifest self-organization and complex behaviors even when the individual strategies of all their agents are simple.

Agents can share knowledge using any agreed language, within the constraints of the system's communication protocol. Example languages are Knowledge Query Manipulation Language (KQML) or FIPA's Agent Communication Language (ACL).

Multi-agent system basics

Multiple agent systems paradigms

Many MAS systems are implemented in computer simulations, stepping the system through discrete "time steps". The MAS components communicate typically using a weighted request matrix, e.g.

```
Speed-VERY_IMPORTANT: min=45 mph,
Path length-MEDIUM_IMPORTANCE: max=60 expectedMax=40,
Max-Weight-UNIMPORTANT
Contract Priority-REGULAR
```

and a weighted response matrix, e.g.

```
Speed-min:50 but only if weather sunny,
Path length:25 for sunny / 46 for rainy
Contract Priority-REGULAR
note - ambulance will override this priority and you'll have to wait
```

A challenge-response-contract scheme is common in MAS systems, where

```
First a "Who can?" question is distributed.
```

```
 Only the relevant components respond: "I can, at this price".
 Finally, a contract is set up, usually in several more short
communication steps between sides,
```

also considering other components, evolving "contracts", and the restriction sets of the component algorithms.

Another paradigm commonly used with MAS systems is the pheromone, where components "leave" information for other components "next in line" or "in the vicinity". These "pheromones" may "evaporate" with time, that is their values may decrease (or increase) with time.

Properties

MAS systems, also referred to as "self-organized systems", tend to find the best solution for their problems "without intervention". There is high similarity here to physical phenomena, such as energy minimizing, where physical objects tend to reach the lowest energy possible, within the physical constrained world. For example: many of the cars entering a metropolis in the morning, will be available for leaving that same metropolis in the evening.

The main feature which is achieved when developing multi-agent systems, if they work, is flexibility, since a multi-agent system can be added to, modified and reconstructed, without the need for detailed rewriting of the application. These systems also tend to be rapidly self-recovering and failure proof, usually due to the heavy redundancy of components and the self managed features, referred to, above.

The study of multi-agent systems

The study of multi-agent systems is "concerned with the development and analysis of sophisticated AI problem-solving and control architectures for both single-agent and multiple-agent systems." Topics of research in MAS include:

- agent-oriented software engineerin
- organization
- communication
- negotiation
- multi-agent learning
- scientific communities
- dependability and fault-tolerance

Consensus dynamics

Consensus dynamics or **agreement dynamics** is an area of research lying at the intersection of systems theory and graph theory. A major topic of investigation is the agreement or consensus problem in multi-agent systems that concerns processes by which a collection of interacting agents achieve a common goal. Networks of agents that exchange information to reach consensus include: physiological systems, gene networks, large-scale energy systems and fleets of vehicles on land, in the air or in space. The agreement protocol or consensus protocol is an unforced dynamical system that is governed by the interconnection topology and the initial condition for each agent. Other problems are the rendezvous problem, synchronization, flocking, formation control. One solution paradigm is distributed constraint reasoning

Cooperative distributed problem solving

Cooperative Distributed Problem Solving is a network of semi-autonomous processing nodes working together to solve a problem, typically in a multi-agent system. That is concerned with the investigation of problem subdivision, sub-problem distribution, results synthesis, optimisation of problem solver coherence and co-ordination. It is closely related to distributed constraint programming and distributed constraint optimization.

Aspects of CDPS

- Neither global control nor global data storage - no individual CDPS problem solver (agent) has sufficient information to solve the entire problem.
- Control and data are distributed
- Communication is slower than computation, therefore:
    - Loose coupling between problem solvers
    - Efficient protocols (not too much communication overhead)
    - problems should be modular, course grained
- Any unique node is a potential bottleneck
    - Organised behaviour is hard to guarantee since no one node has the complete picture

Contract Net Protocol

Contract Net Protocol (CNP) is a well known task sharing protocol that is used for task allocation in multi-agent systems and consists of a collection of nodes or software agents that form the contract net. Each node on the network can, at different times or for different tasks be a manager or a contractor.

When a node gets a composite task (or for any reason cannot solve the present task) it breaks the problem down into sub-tasks (If possible) and announces the sub-task to the contract net acting as a manager. Bids are then received from potential contractors which the winning contractor(s) are awarded the job(s).

The Contract Net

Task distribution is viewed as a kind of contract negotiation and happens in 5 stages.

1. Recognition
2. Announcement
3. Bidding
4. Awarding
5. Expediting

Recognition

An agent recognises it has a problem that it wants help with. The agent has a goal, and either:

- Releases it cannot achieve the goal in isolation - does not have the capability to fulfil the goal
- Realises it would prefer not to achieve the goal in isolation - typically because of solution quality, deadline, etc.

Announcement

The agent with the task sends out an announcement of the task which includes a specification of the task to be achieved. The specification must encode:

- Description of the task itself
- Any constraints
- Meta-task information

Bidding

Agents that receive the announcement decide themselves whether they should bid for the task. Factors that are taken into consideration are:

- The agent must decide whether it is capable of the expecting task
- The agent must determine the quality constrains and the price information (if relevant)

Awarding

Agents that send the task announcement must choose between the received bids and decide who to award the contract to. The result of this process is communicated to agents that submitted a bid.

Expediting

The successful contractor then expedites the task. This may involve the generation of further contract nets in the form of sub-contracting to complete the task.

Example uses of Contract Net Protocol

An electronic market place for buying and selling goods. For example a system where a user could specify the goods that they want as well as a maximum price that they are willing to pay. The agent programs then would find other user(s) willing to sell the goods within the desired price range. The user with the lowest price would then be selected to fulfill the contract. Other constraints could be applied such as delivery time and the location of the goods.

Claytronics

Claytronics is an abstract future concept that combines nanoscale robotics and computer science to create individual nanometer-scale computers called claytronic atoms, or catoms, which can interact with each other to form tangible 3-D objects that a user can interact with. This idea is more broadly referred to as programmable matter. Claytronics has the potential to greatly affect many areas of daily life, such as telecommunication, human-computer interfaces, and entertainment.

Current research

Current research is exploring the potential of modular reconfigurable robotics and the complex software necessary to control the "shape changing" robots. "Locally Distributed Predicates or LDP is a distributed, high-level language for programming modular reconfigurable robot systems (MRRs)". There are many challenges associated with programming and controlling a large number of discrete modular systems due to the degrees of freedom that correspond with each module. For example, reconfiguring from one formation to one similar may require a complex path of movements controlled by an intricate string of commands even though the two shapes differ slightly.

In 2005, research efforts to develop a hardware concept were successful on the scale of millimeters, creating cylindrical prototypes 44 millimeters in diameter which interact with each other via electromagnetic attraction. Their experiments helped researchers verify the relationship between mass and potential force between objects as "a 10-fold reduction in size [which] should translate to a 100-fold increase in force relative to mass". Recent advancements in this prototype concept are in the form of one millimeter diameter cylindrical robots fabricated on a thin film by photolithography that would cooperate with each other using complex software that would control electromagnetic attraction and repulsion between modules.

Today, extensive research and experiments with claytronics are being conducted at Carnegie Mellon University in Pittsburgh, Pennsylvania by a team of researchers which consists of Professors Todd C. Mowry, Seth Goldstein, Ph. D. candidates, graduate and undergraduate students, and researchers from Intel Labs Pittsburgh.

Hardware

The driving force behind programmable matter is the actual hardware that is manipulating itself into whatever form is desired. Claytronics consists of a collection of individual components called claytronic atoms, or catoms. In order to be viable, catoms need to fit a set of criteria. First, catoms need to be able to move in three dimensions relative to each other and be able to adhere to each other to form a three dimensional shape. Second, the catoms need to be able to communicate with each other in an ensemble and be able to compute state information, possibly with assistance from each other. Fundamentally, catoms consist of a CPU, a network device for communication, a single pixel display, several sensors, an onboard battery, and a means to adhere to one another.

Current catoms

The researchers at Carnegie Mellon University have developed various prototypes of catoms. These vary from small cubes to giant helium balloons. The prototype that is most like what developers hope catoms will become is the planar catom. These take the form of 44 mm diameter cylinders. These cylinders are equipped with 24 electromagnets arranged in a series of stacked rings along the cylinder's circumference. Movement is achieved by the catoms cooperatively enabling and disabling the magnets in order to roll along each other's surfaces. Only one magnet on each catom is energized at a time. These prototypes are able to reconfigure themselves quite quickly, with the uncoupling of two units, movement to another contact point, and recoupling taking only about 100 ms. Power is supplied to the catoms using pickup feet on the bottom of the cylinder. Conductive strips on the table supply the necessary power.

Future design

In the current design, the catoms are only able to move in two dimensions relative to each other. Future catoms will be required to move in three dimensions relative to each other. The goal of the researchers is to develop a millimeter scale catom with no moving parts, to allow for mass manufacturability. Millions of these microrobots will be able to emit variable color and intensity of light, allowing for dynamic physical rendering. The design goal has shifted to creating catoms that are simple enough to only function as part of an ensemble, with the ensemble as a whole being capable of higher function.

As the catoms are scaled down, an onboard battery sufficient to power it will exceed the size of the catom itself, so an alternate energy solution is desired. Research is being done into powering all of the catoms in an ensemble, utilizing the catom-to-catom contact as a means of energy transport. One possibility being explored is using a special table with positive and negative electrodes and routing the power internally through the catoms, via "virtual wires."

Another major design challenge will be developing a genderless unary connector for the catoms in order to keep reconfiguration time at a minimum. Nanofibers provide a possible solution to this challenge. Nanofibers allow for great adhesion on a small scale and allow for minimum power consumption when the catoms are at rest.

Software

Organizing all of the communication and actions between millions of sub-millimeter scale catoms requires development of advanced algorithms and programming languages. The researchers and engineers of Carnegie Mellon-Intel Claytronics Research Lab launched a wide range of projects to develop the necessary software to facilitate communication between catoms. The most important projects are developing new programming languages which work more efficiently for claytonics. The goal of a claytronics matrix is to dynamically form three dimensional shapes. However, the vast number of catoms in this distributed network increases complexity of micro-management of each individual catom. So, each catom must perceive accurate position information and command of cooperation with its neighbors. In this environment, software language for the matrix operation must convey concise statements of high-level commands in order to be universally distributed. Languages to program a matrix require a more abbreviated syntax and style of command than normal programming languages such as C++ and Java.

The Carnegie Mellon-Intel Claytronics Research Project has created two new programming languages: Meld and Locally Distributed Predicates (LDP).

Meld

Meld is a declarative language, a logic programming language originally designed for programming overlay networks. By using logic programming, the code for an ensemble of robots can be written from a global perspective, enabling the programmer to concentrate on the overall performance of the claytronics matrix rather than writing individual instructions for every one of the thousands to millions of catoms in the ensemble. This dramatically simplifies the thought process for programming the movement of a claytronics matrix.

Locally distributed predicates (LDP)

LDP is a reactive programming language. It has been used to trigger debugging in the earlier research. With the addition of language that enables the programmer to build operations in the development of the shape of the matrix, it can be used to analyze the distributed local conditions. It can operate on fixed-size, connected groups of modules providing various functions of state configuration. A program that addresses a fixed-size module rather than the entire ensemble allows programmers to operate the claytronic matrix more frequently and efficiently. LDP further provides a means of matching distributed patterns. It enables the programmer to address a larger set of variables with Boolean logic, which enables the program to search for larger patterns of activity and behavior among groups of modules.

Distributed watchpoints

Performance errors for thousands to millions of individual catoms are hard to detect and debug, therefore, claytronics matrix operations require a dynamic and self-directed process for identifying and debugging errors. Claytronics researchers have developed Distributed Watchpoints, an algorithm-level approach to detecting and fixing errors missed by more conventional debugging techniques. It establishes nodes that receive surveillance to determine the validity of distributed conditions. This approach provides a simple and highly descriptive set of rules to evaluate distributed conditions and proves effective in the detection of errors.

Algorithms

Two important classes of claytronics algorithms are shape sculpting and localization algorithms. The ultimate goal of claytronics research is creating dynamic motion in three dimensional poses. All the research on catom motion, collective actuation and hierarchical motion planning require shape sculpting algorithms to convert catoms into the necessary structure, which will give structural strength and fluid movement to the dynamic ensemble. Meanwhile, localization algorithms enable catoms to localize their positions in an ensemble. A localization algorithm should provide accurate relational knowledge of catoms to the whole matrix based on noisy observation in a fully distributed manner.

Future applications

As the capabilities of computing continue to develop and robotic modules shrink, claytronics will become useful in many applications. The featured application of claytronics is a new mode of communication. Claytronics will offer a more realistic sense to communication over long distance called pario. Similar to how audio and video provide aural and visual stimulation, pario provides an aural, visual and physical sensation. A user will be able to hear, see and touch the one communicating with them in a realistic manner. Pario could be used effectively in many professional disciplines from engineering design, education and healthcare to entertainment and leisure activities such as video games.

The advancements in nanotechnology and computing necessary for claytonics to become a reality are feasible, but the challenges to overcome are daunting and will require great innovation. In an interview, December 2008, Jason Campbell, a lead researcher from Intel Labs Pittsburgh said, "my estimates of how long it is going to take have gone from 50 years down to just a couple more years. That has changed over the four years I’ve been working on the project".

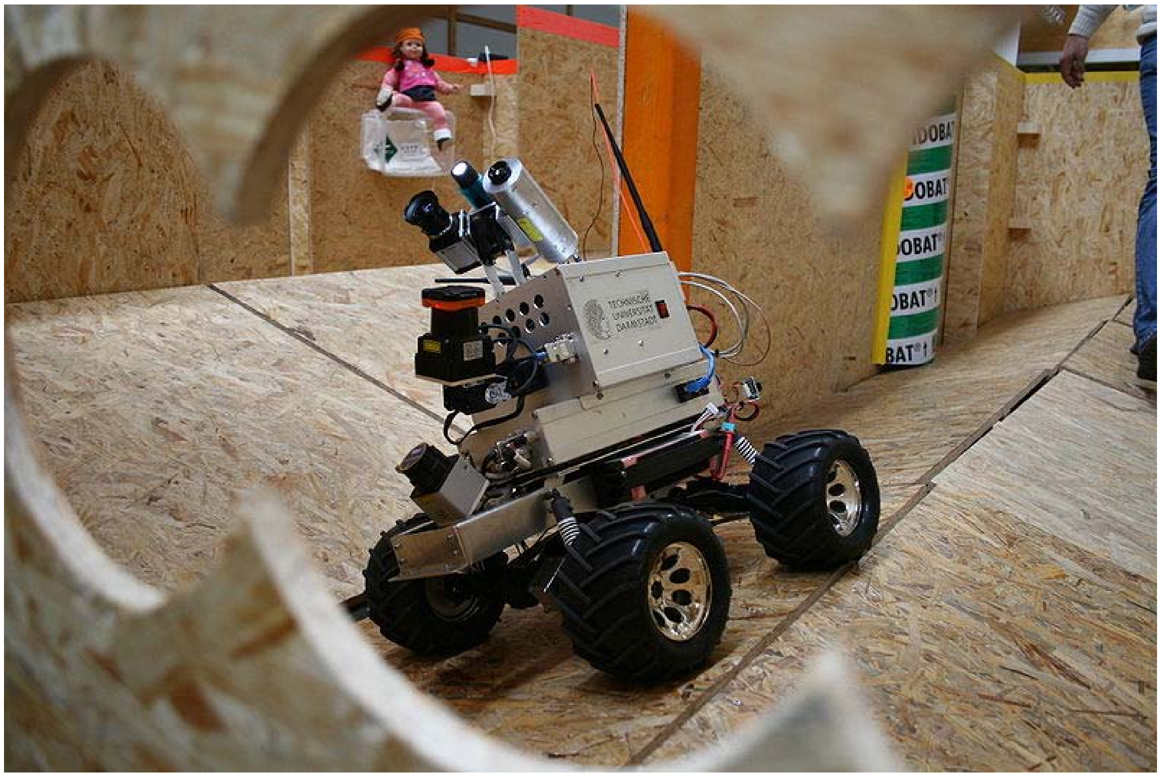
Ant robotics

Ant robotics is a special case of swarm robotics. Swarm robots are simple and cheap robots with limited sensing and computational capabilities. This makes it feasible to deploy teams of swarm robots and take advantage of the resulting fault tolerance and parallelism. Swarm robots cannot use conventional planning methods due to their limited sensing and computational capabilities. Thus, their behavior is often driven by local interactions. Ant robots are swarm robots that can communicate via markings, similar to ants that lay and follow pheromone trails. Some ant robots use long-lasting trails (either regular trails of a chemical substance or smart trails of transceivers), others use short-lasting trails (heat, odor or alcohol), and others even use virtual trails.

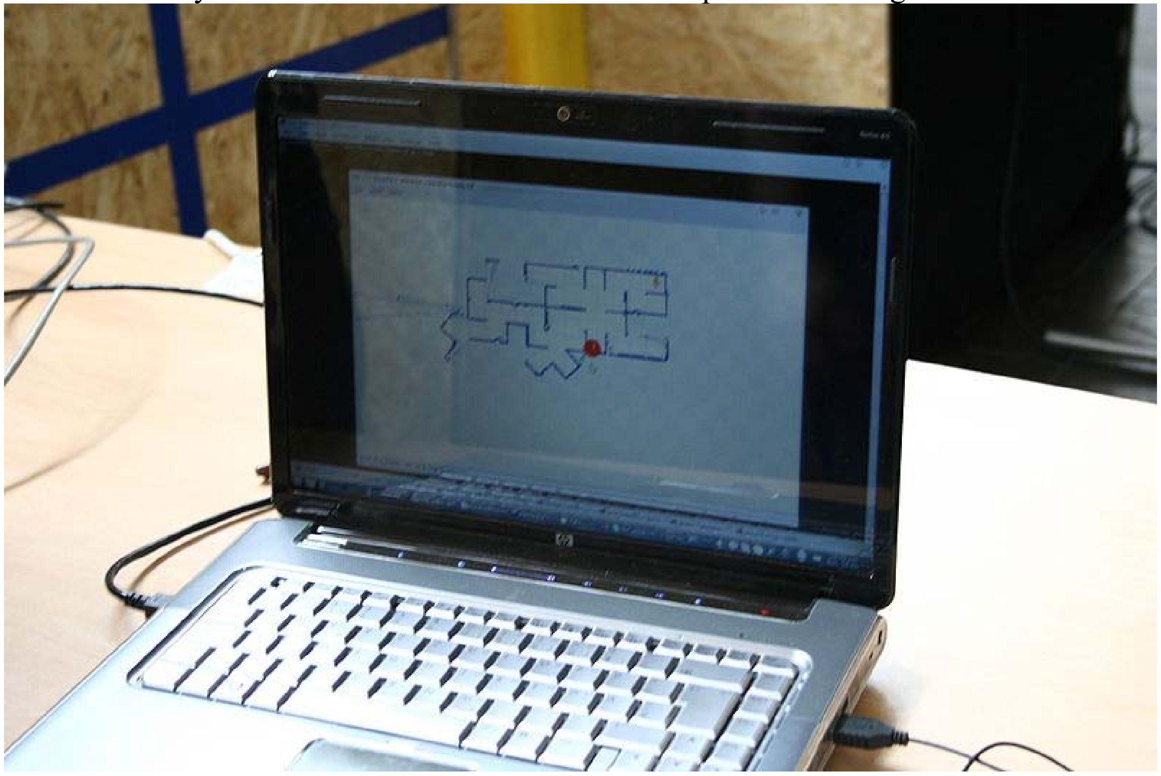
Researchers have developed ant robot hardware and software and demonstrated, both in simulation and on physical robots, that single ant robots or teams of ant robots solve robot-navigation tasks (such as path following and terrain coverage) robustly and efficiently. For example, trails coordinate the ant robots via implicit communication and provide an alternative to probabilistic reasoning for solving the simultaneous localization and mapping problem.

Researchers have also developed a theoretical foundation for ant robotics, based on ideas from real-time heuristic search, stochastic analysis and graph theory. Recently, it was shown that a single ant robot (modeled as finite state machine) can simulate the execution of any arbitrary Turing machine. This proved that a single ant robot, using pheromones, can execute arbitrarily complex single-robot algorithms. However, the result unfortunately does not hold for N robots.

Simultaneous localization and mapping



A robot built by Technische Universität Darmstadt maps a maze using a LIDAR.



A map generated by the Darmstadt Robot.

Simultaneous localization and mapping (**SLAM**) is a technique used by robots and autonomous vehicles to build up a map within an unknown environment (without a priori knowledge) or to update a map within a known environment (with a priori knowledge from a given map) while at the same time keeping track of their current location.

Operational definition

Maps are used to determine a location within an environment and to depict an environment for planning and navigation. They support the assessment of actual location by recording information obtained from a form of perception and comparing it to a current set of perceptions. The benefit of a map in aiding the assessment of a location increases as the precision and quality of the perceptions decreases. However, maps generally represent the state at the time that the map is drawn. This is not necessarily consistent with the state of the environment at the time the map is used.

Complexity of the technical processes of locating and mapping under conditions of errors and of noise do not allow for a coherent solution of both tasks. Simultaneous localization and mapping (SLAM) is a concept to bind these processes in a loop and therefore supports the continuity of both aspects in separated processes. Iterative feedback from one process to the other one enhances the results of both consecutive steps.

Mapping is the problem of integrating the information gathered by a set of sensors into a consistent model and depicting that information as a given representation. It can be described by the **first characteristic question** *What does the world look like?* Central aspects in mapping are the representation of the environment and the interpretation of sensor data.

In contrast to this, localization is the problem of estimating the place (and pose) of the robot relative to a map. In other words, the robot has to answer the **second characteristic question**, *Where am I?* Typically, solutions comprise tracking, where the initial place of the robot is known, and global localization, in which no or just some *a priori* knowledge about the ambiance of the starting position is given.

SLAM is therefore defined as the problem of building a model leading to a new map or repetitively improving an existing map while at the same time localizing the robot within that map. In practice, the answers to the two characteristic questions cannot be delivered independently of each other.

Before a robot can contribute to answer the question of what the environment looks like given a set of observations, it needs to know e.g.

- the robot's own kinematics,
- which qualities the autonomous acquisition of information has,
- from which sources additional supporting observations have been made.

It is a complex task to estimate the robot's current location without a map or without a directional reference. Here, the location is either just the position of the robot or might even include its orientation.

Technical problems

SLAM can be thought of as a chicken or egg problem: An unbiased map is needed for localization while an accurate pose estimate is needed to build that map. This is the starting condition for iterative mathematical solution strategies. In comparison, the atomic orbital model may be seen as a classic approach of how to communicate sufficient results under conditions of imprecise observation.

Beyond, the answering of the two characteristic questions is not as straightforward as it might sound due to inherent uncertainties in discerning the robot's relative movement from its various sensors. Generally, due to the budget of noise in a technical environment, SLAM is not served with just compact solutions, but with a bunch of physical concepts contributing to results.

If at the next iteration of map building the measured distance and direction traveled has a budget of inaccuracies, driven by limited inherent precision of sensors and additional ambient noise, then any features being added to the map will contain corresponding errors. Over time and motion, locating and mapping errors build cumulatively, grossly distorting the map and therefore the robot's ability to determine (know) its actual location and heading with sufficient accuracy.

There are various techniques to compensate for errors, such as recognizing features that it has come across previously, and re-skewing recent parts of the map to make sure the two instances of that feature become one. Some of the statistical techniques used in SLAM include Kalman filters, particle filters (aka. Monte Carlo methods) and scan matching of range data.

Mapping

SLAM in the mobile robotics community generally refers to the process of creating geometrically consistent maps of the environment. Topological maps are a method of environment representation which capture the connectivity (i.e., topology) of the environment rather than creating a geometrically accurate map. As a result, algorithms that create topological maps are not referred to as SLAM.

SLAM is tailored to the available resources, hence not aimed at perfection, but at operational compliance. The published approaches are employed in unmanned aerial vehicles, autonomous underwater vehicles, planetary rovers, newly emerging domestic robots and even inside the human body.

It is generally considered that "solving" the SLAM problem has been one of the notable achievements of the robotics research in the past decades. The related problems of data association and computational complexity are amongst the problems yet to be fully resolved.

Sensing

SLAM will always use several different types of sensors to acquire data with statistically independent errors. Statistical independence is the mandatory requirement to cope with metric bias and with noise in measures.

Such optical sensors may be one dimensional (single beam) or 2D- (sweeping) laser rangefinders, 2D or 3D sonar sensors and one or more 2D cameras.

Recent approaches apply quasi-optical wireless ranging for multi-lateration (RTLS) or multi-angulation in conjunction with SLAM as a tribute to erratic wireless measures.

A special kind of SLAM for human pedestrians uses a shoe mounted inertial measurement unit as the main sensor and relies on the fact that pedestrians are able to avoid walls. This approach called FootSLAM can be used to automatically build floor plans of buildings that can then be used by an indoor positioning system .

Locating

The results from sensing will feed the algorithms for locating. According to propositions of geometry, any sensing must include at least one lateration and (n+1) determining equations for an n-dimensional problem. In addition, there must be some additional a priori knowledge about orienting the results versus absolute or relative systems of coordinates with rotation and mirroring.

Modeling

Contribution to mapping may work in 2D modeling and respective representation or in 3D modeling and 2D projective representation as well. As a part of the model, the kinematics of the robot is included, to improve estimates of sensing under conditions of inherent and ambient noise. The dynamic model balances the contributions from various sensors, various partial error models and finally comprises in a sharp virtual depiction as a map with the location and heading of the robot as some cloud of probability. Mapping is the final depicting of such model, the map is either such depiction or the abstract term for the model.