Achivements(write some content about it)(ask for images)(Home page)

1-First team to build an autonomous walking humanoid made by any undergraduate team in India.

2- https://drive.google.com/drive/folders/1UZ\_HKeWe3IZ7lVkr--dUIxErTZ76oXb7

3-We are the only Indian team which participated in FIRA HUROCUP 2019.

4. Winner of TIC(Techkriti innovation challenge) 2016

5. Best summer project of SNT council 2015"

Automi page (ask for senior auv vs motorsports)

**Current CAD model of the bot Automi -**

<https://drive.google.com/drive/folders/1Ikk1CBz_A_MoDe60fwLv4gH_84duhpJR?usp=sharing>

Areas of Expertise-(write some content about it,Update data in members card)

Kinematics (Inverse and Forward):- **Forward kinematics** refers to the use of the [kinematic](https://en.wikipedia.org/wiki/Kinematic) equations of a [robot](https://en.wikipedia.org/wiki/Robot) to compute the position of the [end-effector](https://en.wikipedia.org/wiki/Robot_end_effector) from specified values for the joint parameters. Inverse kinematics is the use of kinematic equations to determine the motion of a robot to reach a desired position

Trajectory Generation:- Trajectory planning is moving from point A to point B while avoiding collisions over time. This can be computed in both discrete and continuous methods. Trajectory planning is a major area in robotics as it gives way to autonomous vehicles. Trajectory planning is sometimes referred to as motion planning and erroneously as path planning. Trajectory planning is distinct from path planning in that it is parametrized by time. Essentially trajectory planning encompasses path planning in addition to planning how to move based on velocity, time, and kinematics.

Simulations (Gazebo and MATLAB):- Simulations help you understand the design and behavior of your model. Simulate a model interactively by clicking the Run button in the Simulink Toolstrip, or programmatically using functions like sim and set\_param in the MATLAB Command Window or a MATLAB script. Gazebo is a powerful robot simulator used by industry and academia that calculates physics, generates sensor data and provides convenient interfaces. Open source software is lowering the barrier to entry and speeding up progress in robotics

SLAM:- Simultaneous Localisation and Mapping (SLAM), a technology which allows a device to map its environment while positioning itself in it, is a crucial driver for the future of robotics. SLAM software enables the transition from Automated Guided Vehicles (AGVs) to Autonomous Mobile Robots (AMRs) in the industrial space.

Structural Design Analysis (ANSYS):- Ansys offers structural analysis software solutions that enable engineers of all levels and backgrounds to solve complex structural engineering problems faster and more efficiently. With our suite of tools, engineers can perform finite element analyses (FEA), customize and automate solutions for structural mechanics challenges and analyze multiple design scenarios. By using our software early in the design cycle, businesses can save costs, reduce the number of design cycles and bring products to market faster.

Fusion 360:- Fusion 360 is a cloud-based 3D modeling, CAD, CAM, CAE, and PCB software platform for professional product design and manufacturing.

* Design and engineer products how you want to ensure aesthetics, form, fit, and function
* Engineer, design, and create anything with comprehensive electronics and PCB design tools
* Save time and money and get quality parts out the door faster

Inventor:- Inventor is computer-aided design (CAD) software developed by Autodesk, many use for **3D mechanical design, simulation, and visualization**. It uses the concept of parametric design, used primarily to create technical drawings for mechanical purposes.

ROS:- **Robot Operating System** (**ROS or ros**) is an open-source [robotics middleware](https://en.wikipedia.org/wiki/Robotics_middleware) suite. Although ROS is not an [operating system](https://en.wikipedia.org/wiki/Operating_system) (OS) but a set of [software frameworks](https://en.wikipedia.org/wiki/Software_framework) for [robot](https://en.wikipedia.org/wiki/Robot) software development, it provides services designed for a heterogeneous [computer cluster](https://en.wikipedia.org/wiki/Computer_cluster) such as [hardware abstraction](https://en.wikipedia.org/wiki/Hardware_abstraction), low-level [device control](https://en.wikipedia.org/wiki/Device_driver), implementation of commonly used functionality, [message-passing between processes](https://en.wikipedia.org/wiki/Inter-process_communication), and package management. Running sets of ROS-based processes are represented in a [graph](https://en.wikipedia.org/wiki/Graph_theory) architecture where processing takes place in nodes that may receive, post, and [multiplex](https://en.wikipedia.org/wiki/Multiplexing) sensor data, control, state, planning, actuator, and other messages. Despite the importance of reactivity and [low latency](https://en.wikipedia.org/wiki/Low_latency) in robot control, ROS is *not* a [real-time operating system](https://en.wikipedia.org/wiki/Real-time_operating_system) (RTOS). However, it is possible to integrate ROS with [real-time](https://en.wikipedia.org/wiki/Real-time_computing) code.[[3]](https://en.wikipedia.org/wiki/Robot_Operating_System#cite_note-3) The lack of support for real-time systems has been addressed in the creation of ROS 2,[[4]](https://en.wikipedia.org/wiki/Robot_Operating_System#cite_note-4)[[5]](https://en.wikipedia.org/wiki/Robot_Operating_System#cite_note-5)[[6]](https://en.wikipedia.org/wiki/Robot_Operating_System#cite_note-6) a major revision of the ROS API which will take advantage of modern libraries and technologies for core ROS functions and add support for real-time code and [embedded system](https://en.wikipedia.org/wiki/Embedded_system) hardware.

Projects(need to update data)(done)

https://docs.google.com/document/d/1vGi7aH4B6ID5JmWSqe9iGOc0dlCYeNpDaAZf-3UBbng/edit?usp=sharing

simulations in excel

3d simulation

Cad model of automi

* **LIBHURO**
  + Library/framework for basic operations needed by our bot.
  + Path planner - objective optimisation based planner in C++ (in-production).
  + Trajectory Generation - Given end point and starting point, generate trajectory along with inputs to motor to traverse the trajectory.. This is done through a MATLAB model of bot.
  + Range sensing using only passive sensors (such as stereo cameras).

* **SLAM**
  + Our implementation consists of 3 modules (in-production):
    1. Tracking - where the bot is in the environment.
    2. Local Mapping - to map out all the objects and walls.
    3. Loop Closure - to recognize a previously visited location
  + Tracking:
    1. ORB based feature detection
    2. Pose estimation
  + Local mapping:
    1. Particle Filter for map building
  + Loop Closure :
    1. When a loop is recognized, the drift in data is accounted for and adjusted.

* **Design Analysis**
  + Structural Analysis of any metallic body part is necessary as it will help us identify any defects in the design of the robot, which can be corrected.
  + The robot has been analysed by taking into consideration all the forces and weights acting on it.
  + Consists of:
    1. Stress
    2. Strain
    3. Deformation
  + We used ANSYS Workbench static structural module.

* **Design:**
  + Hand design - We currently have different hands for doing different tasks. So we are currently working on making a universal hand which can do all the tasks we need the bot to do.
  + Pneumatic arm - We are planning to make a new bigger robot which will be mainly based on pneumatic based actuation. We are currently working on designing an arm for it.

* **Voice Localization:**
  + Arduino based high frequency detection using FFT
  + Bot points in direction of incoming audio

* **Face Recognition:**
  + Face recognition of all team members
  + OpenCV as the primary CV library used
  + Haar Cascade object detection used to identify faces

* **Competition oriented task modules:**
  + Frequent competitions tasks such as Basketball, Marathon, Archery
  + Modules for each consisting of relevant codes
  + For eg - object localization for Basketball, Target detection for Archery etc

# Humanoid Team is a student run organization at one of the premier technical institutes of the country, Indian Institute of Technology, Kanpur. We are a team of highly motivated, enthusiastic, passionate and dedicated students who work towards the ultimate goal of developing Humanoid robots. Our team comprises of students from a spectrum of different departments and years (i.e. First years to Post Graduates) who are passionate to pursue, explore, learn. We began our venture in the year 2011, by the collective efforts of a group of students with a strong urge to complement the theoretical knowledge imparted in classrooms, emphasizing on the practical aspects of engineering. Their idea to develop an automotive culture revolutionized the institute and SAE IIT Kanpur started with an initial strength of over 150 student members.The competitions we participate in are highly prestigious and are judged by the professionals who command respect in the automotive industry in order to showcase our technical prowess, skills and hone our knowledge pertaining to real life engineering. By becoming a part of SAE IIT Kanpur, students embark upon the path that will nurture priceless practical skills in their respective fields of both engineering and business, enriching their college experience and equipping them with the knowledge that will allow them to prosper in their chosen career.

# About us and achievements