



**AutoArea**

**Area Annotation Software for AutoCAD drawings**

**Phase - 2**

# Team

Gautam Garani

Dhaksh S Kumar



# Introduction

## Abstract

- The AutoArea project presents an **automated** solution for area calculation and annotation within AutoCAD drawings by integrating **AutoLISP** and **Python**.
- The system leverages AutoCAD **event reactors** to trigger external Python scripts during predefined drawing events, enabling accurate and consistent **area computation** with minimal user intervention.
- Emphasis is placed on maintaining drawing integrity by avoiding unnecessary modifications to the DWG file.

# Project Overview

## Problem Statement

- In conventional AutoCAD workflows, area calculation and annotation are predominantly manual processes. These tasks are time-consuming.
- While AutoCAD provides basic tools for area measurement, they often require explicit user action and lack automation, customization, and integration with external logic.
- A reliable, non-intrusive automation mechanism is therefore required.

## Literature Insights

- Limited work has been documented on tightly coupled AutoLISP–Python systems that execute automatically.
- *No documented work provides for a **secure drawing environment rid of third-party applications.***

# Core Objectives

## Main Objective

- To design and implement an automated system for area calculation and annotation in AutoCAD.
- To integrate AutoLISP reactors with external Python scripts for event-driven execution.
- To ensure that automation does not unnecessarily *modify* or *dirty* the drawing file.

# Fields of Relevance

The model finds relevance in the following fields:

1. **Construction and Architecture:** Automated area annotation helps to instantaneously spot architectural flaws.
2. **Defence and Aerospace:** Precise and secure area-annotations are crucial in the sector.

# Methodology

## System Architecture Design

- The AutoArea system follows an event-driven architecture combining AutoCAD's native AutoLISP environment with external Python processing.
- AutoLISP reactors are used to monitor specific drawing events (such as save operations) and act as the trigger mechanism.
- Upon activation, the reactor **invokes a Python script** through the system shell without blocking the AutoCAD interface.
- The Python module performs area calculations and annotation logic independently and, where required, communicates results back to AutoCAD in a controlled manner.

# Methodology

## Tools and Libraries

- **AutoLISP:** Used for AutoCAD integration, event reactors, entity selection, and triggering external processes.
- **AutoCAD API (imported as pyautocad):** Provides access to drawing entities, geometry, and annotation mechanisms.
- **Python:** Handles the computational logic and processing.



# Results



A view of the results generated by AutoArea

# Code Snippets

AutoArea / AutoRun\_AutoArea.lsp 

**G** GautamGarani Add AutoRun for on-save reaction

Code Blame 26 lines (18 loc) · 671 Bytes 

```
1 ; program to automatically run python AutoArea script on save
2 ;-----
3
4 (defun AutoArea:run_py (reactor params)
5   (startapp "\"C:/Users/GAUTAM GARANI/AppData/Local/Programs/Python/Python314/python.exe\""
6     "\"C:/Users/GAUTAM GARANI/Documents/Code/Git/AutoArea/main.py\"")
7 )
8
9
10
11 ; Reactor should NOT run the python program multiple times:
12 ; Reactor is created if it does not exist, and stored into AutoArea_SaveReactor so that it is NOT created again and again
13
14 (if (not AutoArea_SaveReactor)
15   (setq AutoArea_SaveReactor
16     (vlr-dwg-reactor
17       ""
18       '(:vlr-saveComplete . AutoArea:run_py))
19   )
20 )
21
22
23 (princ)
24
25
26
```

A view of the AutoLISP code written by the team

# Conclusion and Comparative Analysis

External Area Calculation	AutoArea
<ul style="list-style-type: none"><li>• Lengthy and time-consuming.</li><li>• Offers low flexibility and clutters drawings.</li></ul>	<ul style="list-style-type: none"><li>• Significantly reduces user effort and eliminates repetitive tasks.</li><li>• <b>AutoArea</b> operates <i>directly within the AutoCAD workflow</i>, providing <b>real-time automation</b> while <b>preserving drawing integrity</b>.</li></ul>

# Future Work

The model proposes an extension to **support additional geometric properties** such as perimeter, volume, and centroid annotation.