**Introduction to Online and Offline Programming**

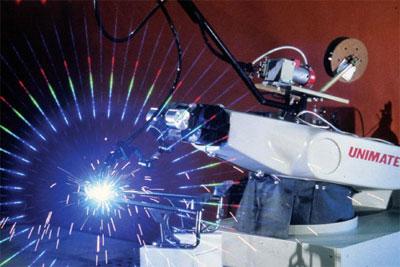
Almost every robot has its own programming language. A robot’s programming language is how a robot knows exactly what to do. Online programming is an easy, standard method for programming robots [40]. No special programming abilities or knowledge is required for online programming. The main thing that separates online programming from offline programming is hinted at in the name. Online programming requires direct interaction and access to the robot of interest in order to give the robot instructions. Offline programming is a method of programming where instructions can be created and given without direct access to the robot. Offline programming can be created on a computer and implemented later. Existing code, data, or programs can be given to a robot. Online programming is less practical in terms of convenience, but it is efficient in terms of kinetic specification. Online programming using a mix of teach-in and playback programming. The operator manually moves the robot’s components and carefully calculates the movements the operator wants the robot to make. The movements are then saved locally to the robot's memory. The movements may not be entirely precise, but the programmer would have a visual example of exactly what the robot has been told to do.

One major flaw in online programming is that a robot’s online program can be hard to transfer, translate, modify, or up-keep. In contrast, offline programming is easy to document and maintain. The robot does not need to be present in order for code to be written. When using offline programming, the robot’s movements are more precise and sharp, and the robot can still function while new program is being created. Using offline programming, it is easier to create code that can be used in multiple different robots. Offline programming is more convenient for mass production of robots. A universal code can be used in multiple robots. However, offline programming uses text-based programming language, like Python, C++, C, or Java. Someone with extensive programming knowledge and experience is required to program a robot offline. The major downside to offline programming is that special, expensive equipment may be required to get the job done. Offline programming is time-efficient, but not as cost-efficient in the way that online programming is.

**Background of Online and Offline Programming**

George Devol’s Unimate was the first “real” industrial robot. The Unimate was a online programmed mechanical arm. The Unimate did not have a specific robotic programming language. It used the teaching and playback method of programming [33][34]. Users would manually move the robot’s arm, program the joints, and then the program would be played back sending signals to the robot to go back to its pre-programmed position. This method of programming made it easy for the Unimate to perform repeated actions making the Unimate the first mass-produced robot. Most early robots were online programmed robots. There were not as many programming languages as there are today, and the ones that existed were hard to translate from robot to robot. For this period of time and the state of robotics, online programming was the most efficient programming method.

The original goal of offline programming was to help eliminate errors and create perfect tasks for industrial robots. Earlier computers lacked many features that modern-day computers have. For example, computers in the 1960s did not have the processing power to digitally simulate robot’s functions [26]. It was not until the 1980s when engineers began to integrate CAD into robotics programming software that offline programming became truly effective. This revolution created virtual simulation spaces for engineers to test the robot before actually building it. Offline programming, or OLP, reduced production time and cost. Engineers could effectively rule out any imperfections and test robots without directly calculating each and every movement. Starting in the 1990s, industries like healthcare, aerospace, and electronics have all benefited from OLP[38]. Robots can be programmed and tested quickly and efficiently ruling out critical errors and saving the entire industry time.



**Figure 1**| The very first industrial robot, the UNIMATE 1900 series. The UNIMATE 1900 was one of the first online prorgammed robot.

**Comparative Analysis of Online and Offline Programming**

Decision-making between the online and offline methodologies must arise in this rapidly changing environment of robotics. This would be determined depending on the exact nature of the task requirement, the operational environment, and the skill set of the personnel involved. Understanding the strengths and weaknesses of each approach will help in optimized robotic applications to truly serve the needs of an organization.

**Flexibility vs. Accuracy**

Online programming allows operators to take robots out of production and into a programming or teaching mode. This real-time programming capability enables operators to create or modify programs on the fly. One common method employed in online programming is teach pendant programming. This approach uses a handheld device connected to the robot, enabling operators to manually guide the robot through desired positions. Remarkably, around 90% of industrial robots are programmed using this method. Another technique is lead-through programming, where operators physically guide the robot arm through specified waypoints. This intuitive method is particularly beneficial for collaborative robots, making it accessible even to users who are new to automation[1]. For example, with robots like the RO1, operators can seamlessly integrate the robot into their setups and modify its tasks without needing in-depth coding knowledge. This flexibility allows for quick adjustments when unforeseen obstacles arise, optimizing productivity and enhancing safety[2]. As for programming, the robot can be moved using different coordinate systems [10].

However, the flexibility offered by online programming does come with trade-offs, particularly regarding accuracy. Programming robots online can be labor-intensive[3]. Human intervention during operation increases the risk of errors, which can compromise both the performance and safety of the robotic system[5].

On the other hand, offline programming allows the development of robot programs without the need for the robot to be operational. This method utilizes simulation software to model the robot's movements within a virtual environment, enabling thorough testing before actual deployment. Since the robot remains idle during programming, the risk of accidents or damage to equipment is significantly reduced[5]. Additionally, offline programming allows robots to be programmed while they engage in other tasks, leading to a noticeable decrease in downtime[6].

That said, offline programming may lack the flexibility needed to adapt to real-time changes. Once a program is created, making necessary modifications requires additional time for simulation and validation, which could cause delays in fast-paced environments[7]. Nevertheless, this method shows promise for industrial applications due to its learnability, reduced robot downtime, and versatility across various robot types and sizes.

**Mainstream Application of On-Line Programming**

Online programming allows robots to be programmed or adjusted in real-time during operation, making it a critical feature for industries that require flexibility and efficiency. Leading robotics manufacturers such as Universal Robots, KUKA, FANUC, ABB, and Yaskawa have integrated this capability into their mainstream models, enhancing productivity and reducing downtime in sectors like manufacturing, automotive, and electronics (Universal Robots, n.d.; KUKA Robotics, n.d.; FANUC Robotics, n.d.; ABB Robotics, n.d.; Yaskawa Motoman, n.d.).

Universal Robots (UR Series) are collaborative robots used in assembly, material handling, and quality control. Their teach pendant allows operators to program tasks without stopping the robot, making them accessible and easy to use even for non-experts (Universal Robots, n.d.). KUKA’s KR Series, known for their precision in heavy-duty tasks such as welding and material handling, uses the KUKA smartPAD for real-time adjustments, enabling seamless workflow in industries like automotive (KUKA Robotics, n.d.).

FANUC robots, including the LR Mate and CRX series, support online programming through a teach pendant. This capability is crucial for dynamic environments, such as electronics manufacturing, where tasks may frequently need to be adjusted (FANUC Robotics, n.d.). ABB Robotics' IRB Series, equipped with the ABB FlexPendant, allows for real-time programming, making them ideal for welding, painting, and other high-precision tasks (ABB Robotics, n.d.). Lastly, Yaskawa Motoman robots, using DX200 and YRC1000 controllers, are versatile for welding, material handling, and assembly, with real-time adjustments that help maintain operational efficiency (Yaskawa Motoman, n.d.).  
 Online programming improves flexibility by enabling task adjustments without halting operations, reducing downtime, and improving productivity. Robots can quickly adapt to changing production demands, and their user-friendly interfaces make them accessible to operators with varying skill levels. This feature is especially important in industries where quick changes and real-time precision are critical to maintaining efficiency (RobotWorx, n.d.; Automation World, n.d.).

Robots from Universal Robots, KUKA, FANUC, ABB, and Yaskawa demonstrate the transformative power of online programming in industrial automation. By allowing real-time adjustments, these robots maximize uptime, streamline workflows, and improve flexibility, making them indispensable in modern industries (Universal Robots, n.d.; KUKA Robotics, n.d.; FANUC Robotics, n.d.; ABB Robotics, n.d.; Yaskawa Motoman, n.d.).

**Critical Analysis of Offline Programming Robotics**  
 This section will go over the analysis of different examples of Robots that was made with Offline Programming, but first, here is the main ideas of Offline Programming. The main thing about Offline Programming is “a powerful tool in robotics. Creating a digital twin of your robot in OLP helps you experiment and perfect tasks. This leads to smoother real-world operations.” (Djarma.kin, & Admin. 2024). “allowing program industrial robots without stopping production.” (2024) This tide with the software of the machine, allowing to do different types of tasks. Some advantages for Offline programming includes “Reduces robot downtime since programming is completed offline without the robot present, Best for complex programs as saves time from manually entering steps, and Simulation ensures accurate programming”, but for the disadvantages, they are “Requires additional software and a computer and May require additional training for programmers (Robots Done Right). As well as Complex offline programming can take many hours or days to create in this way and so the robot can be down and not generate income for considerable time periods(Berge 2023).

Offline Programming Software “provides you with a software model of the actual robot cell on the production floor. The software comes with a working, moving 3D model of the equipment, which has the same kinematic characteristics of the system.” (Berge 2023). So, one of the task they can do is call Robot Machining, allowing “Increased accuracy of industrial robots and more intuitive offline programming” (Owen-Hill 2023). Some other examples of Offline Programming would be Robot Painting, Robot Spot Welding, 3d printing, Conveyor Picking, Automated Inspection, Drawing, and Integrating an External Axis. (Owen-Hill 2018).

The main Offline Programming would used at a welding or a factory, because robots do not have downtime they product produce way faster than humans most of the time. Research shows that: “Programming time can be reduced up to 80% and robot utilization increased by as much as 95%, boosting programmer productivity and cutting cell downtime”(2023). They also are safe to use if the task is consider dangerous to human, “Reduced risk of accidents and injuries”(2023). It also have “Smooth transition from idea to production” (Owen-Hill 2023).  
 “Offline Programming or OLP solutions are therefore sought after by industries looking to utilize robotic solutions. Due to the advancement of simulation technology and offline programming software, it's becoming faster, more reliable, and more efficient to use OLP solutions than the traditional approach.” (2021). “Critical to the offline programming strategy is application suitability”(2004). Companies wanted an all-in-one solution that could integrate with existing hardware. Rather than a series of piecemealed solutions to get the ideal setup, manufacturers and integrators sought out an integrated solution that works with various applications, especially in a high-mix, low-volume operation( Robotmaster 2021). Offline programming is used more often for a wide range of device types and packages, independent of programming volumes” (2004).  
 Some examples of Offline Programming would be “Afrit, a South African manufacturer of large trailers, implemented OLP” (I. M. 2023). So overall, “Using offline programming, manufacturers can build high-functioning, rapidly iterating factories that can adapt to the ever-changing demands of modern-day production”(2024).

The skill requirements for online and offline programming differ significantly. Online programming offers notable accessibility. The intuitive nature of teach pendant systems allows operators without extensive programming backgrounds to manage robotic tasks effectively. This ease of use reduces training costs and enhances operational flexibility, making online programming a preferred choice for organizations looking to minimize investments in specialized training. However, despite its user-friendly interface, online programming can pose challenges. Operators must be adequately trained to recognize potential issues during operation. The need for quick decision-making demands a certain level of expertise. While this investment in training may increase operational costs, it is often justified by the enhanced accuracy and efficiency that skilled programmers contribute to complex projects[6].

In contrast, offline programming requires operators to engage with simulation tools that demand both technical skills and a solid understanding of the physical processes involved in robotic tasks. This complexity can create a steeper learning curve, necessitating a more significant upfront investment in human resources. However, offline programming's ability to simulate and test robot movements without interrupting production can significantly reduce programming time and improve the intuitiveness of human-robot interaction[3] [9]. Software solutions like Robotmaster streamline the programming process, enabling task experts to efficiently program robots from simple to complex tasks, maximizing their capabilities while minimizing barriers to implementation[9].

Ultimately, blending the advantages of both online and offline programming approaches can provide a more holistic solution. By employing a framework that utilizes augmented and virtual reality, users can select programming techniques that best suit their needs at different stages. This integration can drastically reduce programming time while enhancing user experience[8].

**Cost Implications**

From a financial perspective, online programming can be more cost-effective initially due to lower training requirements and the ability to make immediate adjustments during operation. However, the labor-intensive nature of online programming may lead to increased operational costs over time. Additionally, the potential for human error during programming can result in costly mistakes, negatively impacting both performance and safety[6] [7].

Conversely, while offline programming may require a higher initial investment in training and software, its long-term benefits often outweigh these costs. By facilitating comprehensive testing and validation before deployment, offline programming can significantly reduce the risk of errors that could lead to expensive downtime or rework. The advantages of offline programming are evident in its ability to enhance productivity and operational efficiency, allowing for more precise and reliable robotic applications [3] [9].

#### **Other Comparisons:**

| **Aspect** | **Online Programming**   |  | | --- | | **Offline Programming**   |  | | --- |  |  | | --- | |
| --- | --- | --- | --- | --- | --- |
| **Adaptability to Changes**   |  | | --- |  |  | | --- |  |  | | --- | | Highly adaptable; can adjust to unexpected changes during operation. | Less adaptable; changes require re-running simulations and validations. |
| **User Engagement** | Encourages direct interaction, fostering better understanding of robotic tasks.   |  | | --- |  |  | | --- | | More remote; users may become less familiar with the physical robot's nuances over time. |
| **Collaboration** | Facilitates real-time collaboration among operators and engineers. | | Collaboration is limited to planning stages; team members may not see immediate results. | | --- |  |  | | --- | |
| **Integration with AI**   |  | | --- |  |  | | --- | | | Can leverage AI in real-time to enhance learning and decision-making. | | --- |  |  | | --- | | | Often involves pre-defined algorithms; AI integration is typically done during simulation. | | --- |  |  | | --- | |
| **Future Trends**   |  | | --- |  |  | | --- | | | Increasing use of augmented reality (AR) to improve online programming interfaces. | | --- |  |  | | --- | | Growing interest in integrating machine learning to optimize simulation-based programming. |
| **Maintenance Needs** | | Requires ongoing maintenance to ensure consistent performance in changing environments. | | --- |  |  | | --- | | Less frequent maintenance needed after initial deployment; focus on fine-tuning and updates. |

**Conclusion**

In summary, the decision between online and offline programming methodologies should be informed by a careful analysis of specific operational needs, environmental conditions, and available resources. Online programming offers flexibility and accessibility, making it ideal for dynamic environments. In contrast, offline programming excels in accuracy and detailed planning for complex tasks. By understanding the strengths and limitations of each approach, organizations can select the most effective programming strategy for their robotic applications, ultimately enhancing productivity and efficiency.

References

1. Robots Done Right. (2021). *Online robot programming vs. offline robot programming*. Retrieved from<https://robotsdoneright.com/Articles/online-robot-programming-vs-offline-robot-programming.html>
2. Standard Bots. (n.d.). *Standard Bots: Online vs. offline programming*. Retrieved from<https://standardbots.com/>
3. Fraunhofer. (2021). *Online vs. offline programming: The choice is yours*. Retrieved from<https://publica.fraunhofer.de/entities/publication/d0925045-fd62-4f66-b52d-41c311ec5caf/details>
4. Artiminds. (n.d.). *Robot programming software*. Retrieved from<https://www.artiminds.com/robot-programming-software/>
5. Octopuz. (n.d.). *Robotics: The future of automation*. Retrieved from<https://octopuz.com/>
6. MDPI. (2023). *The importance of offline programming*. Retrieved from<https://www.mdpi.com/2218-6581/13/3/35>
7. arXiv. (2021). *The role of robot programming in Industry 4.0*. Retrieved from<https://arxiv.org/abs/2107.01884>
8. Universal Robots. (n.d.). *Universal Robots: Online and offline programming*. Retrieved from<https://www.universal-robots.com/>
9. Robotmaster. (n.d.). *The advantages of online programming*. Retrieved from<https://www.robotmaster.com/en>
10. Automate UK. (n.d.). *Robot programming methods*. Retrieved from<https://www.automate-uk.com/our-associations/bara/expert-advice/robots-explained/robot-programming-methods/>
11. Standard Bots. (n.d.). *What is offline robot programming (OLP)? A complete guide*. Retrieved from<https://standardbots.com/blog/offline-robot-programming>
12. Berge, J. (2023, November 14). *Offline programming of welding robots*. *The Fabricator*. Retrieved from<https://www.thefabricator.com/thefabricator/article/automationrobotics/offline-programming-of-welding-robots>
13. Heley, J. (2023, December 18). *Robot offline programming*. *A3L*. Retrieved from<https://a3l.uk.com/robot-offline-programming/>
14. Owen-Hill, A. (2018, April 13). *Top 10 tasks for offline programming*. *RoboDK Blog*. Retrieved from<https://robodk.com/blog/tasks-offline-programming/>
15. Visual Components. (2023, June 15). *Robot offline programming (OLP): The complete guide (with examples)*. Retrieved from<https://www.visualcomponents.com/blog/offline-robot-programming-olp-the-complete-guide-with-examples/>
16. Industry EMEA. (2023, June 26). *A complete guide to offline robot programming (OLP) from visual components*. Retrieved from<https://www.industryemea.com/market-overview/69426-offline-robot-programming-olp-the-complete-guide>
17. Engineering, E. (2004, September 1). *Inline vs. offline device programming*. *Electronic Design*. Retrieved from<https://www.electronicdesign.com/home/article/21200931/inline-vs-offline-device-programming>
18. Robotic Simulation Services. (n.d.). *12 things to consider when exploring offline robot programming software solutions*. Retrieved from<https://roboticsimulationservices.com/12-things-to-consider-when-exploring-offline-robot-programming-software-solutions/>
19. Hypertherm Inc. (2021, November 15). *Big trends in offline programming software for robots*. *Automate*. Retrieved from<https://www.automate.org/robotics/news/big-trends-in-offline-programming-software-for-robot>
20. Owen-Hill, A. (2023, August 28). *Easier and faster offline programming*. *RoboDK Blog*. Retrieved from<https://robodk.com/blog/industrial-robot-offline-programming/>
21. Universal Robots. (n.d.). *Collaborative robots (Cobots) for manufacturing automation*. Retrieved from<https://www.universal-robots.com/>
22. KUKA Robotics. (n.d.). *KUKA Robotics industrial robots and automation solutions*. Retrieved from<https://www.kuka.com/>
23. FANUC Robotics. (n.d.). *FANUC automation: Robots for every industry*. Retrieved from<https://www.fanuc.eu/>
24. ABB Robotics. (n.d.). *ABB Robotics and discrete automation*. Retrieved from <https://global.abb/robotics>
25. Yaskawa Motoman. (n.d.). *Yaskawa: Robotic automation solutions*. Retrieved from<https://www.motoman.com/>
26. RobotWorx. (n.d.). *Introduction to online programming for industrial robots*. Retrieved from <https://www.robots.com/articles/online-vs-offline-robot-programming>
27. Automation World. (n.d.). *The benefits of online robot programming*. Retrieved from <https://www.automationworld.com/home/article/13313767/online-vs-offline-programming-for-robotic-automation>
28. FANUC Corporation. (n.d.). *FANUC teach pendants and real-time programming*. Retrieved from <https://www.fanuc.co.jp/en/>
29. KUKA. (n.d.). *Understanding the KUKA smartPAD for real-time adjustments*. Retrieved from <https://www.kuka.com/-/media/kuka-downloads/>
30. Yaskawa. (n.d.). *Advanced control and programming with Yaskawa DX200*. Retrieved from<https://www.yaskawa.com/>
31. Admin. (2024, July 4). *A Guide to Offline Programming (OLP) in Robotics*. Augmentus. Retrieved from <https://www.augmentus.tech/blog/offline-programming-robotics/>
32. *A history of robot programming languages*. (2023, June 20). Control Engineering. Retrieved from <https://www.controleng.com/articles/a-history-of-robot-programming-languages/>
33. Association for Advancing Automation Robotics. (2024). *Unimate - The First Industrial Robot: A tribute to Joseph Engelberger*. Automate. Retrieved September 30, 2024, from

<https://www.automate.org/robotics/engelberger/joseph-engelberger-unimate#:~:text=Revolutionizing%20manufacturing%20the%20world%20over,%20the%20Unimate%20was%20the%20very>

1. Team, R. (2023b, December 19). *Unimate*. ROBOTS: Your Guide to the World of Robotics. Retrieved from <https://robotsguide.com/robots/unimate/>
2. Lika, K. (2024, July 10). Robot Programming Language: 5 options explored. *Bota Systems*. Retrieved from <https://www.botasys.com/post/robot-programming-language#:~:text=Explore%20the%20top%205%20programming%20languages%20for%20robotics,%20from%20Python>
3. OCTOPUZ Offline Robot Programming Software. (2020, May 28). *What is Offline Robot Programming (OLRP)?* [Video]. YouTube. <https://www.youtube.com/watch?v=aeGLn8JTvzc>
4. Abebe, R., & Gopal, M. (2023). Optimization and simulation of arc welding robot parameters based on offline programming. *Materials Today Proceedings*. Retrieved from <https://doi.org/10.1016/j.matpr.2023.02.291>
5. Thoo, Y. J., Maceiras, J., Abbet, P., Racca, M., Girgin, H., & Calinon, S. (2021, July 5). *Online and offline robot programming via augmented reality workspaces*. arXiv.org. Retrieved from <https://arxiv.org/abs/2107.01884#:~:text=View%20a%20PDF%20of%20the%20paper%20titled%20Online%20and%20Offline>
6. Engineering Technology Simulation Learning Videos. (2015, June 5). *Robotics: Online Programming - Teach pendant & Lead-through* [Video]. YouTube. Retrieved from <https://www.youtube.com/watch?v=EA6pWwNI_wg>
7. McCrea, N. (2018, October 16). *An introductory robot programming tutorial*. Toptal Engineering Blog. Retrieved from <https://www.toptal.com/robotics/programming-a-robot-an-introductory-tutorial>

| **Name:** | **Papers:** |
| --- | --- |
| Akiya Garner | **Introduction and Background of Online and Offline Programming** |
| Jordyn Johnson | **Mainstream Application of On-Line Programming** |
| Camron Gayle | **Critical Analysis of Offline Programming Robotics** |
| Dipin Dawadi | **Comparative Analysis of Online and Offline Programming and Conclusion** |