A Simple and Efficient Approach for Extracting Object Hierarchy in Image Data

Saravit Soeng[1], Vungsovanreach Kong[2], Munirot Thon[3], Wan-Sup Cho[4], Tae-Kyung Kim[5]*

Department of Big Data, Chungbuk National University, Cheongju, South Korea[1,2,3,4,5]
Department of Management Information Systems, Chungbuk National University, Cheongju, South Korea[5]

***Abstract*—An object hierarchy in images refers to the structured relationship between objects, where parent objects have one or more child objects. This hierarchical structure is useful in various computer vision applications, such as detecting motorcycle riders without helmets or identifying individuals carrying illegal items in restricted areas. However, extracting object hierarchies from images is challenging without advanced techniques like machine learning or deep learning. In this paper, a simple and efficient method is proposed for extracting object hierarchies in images based on object detection results. This method is implemented in a standalone package compatible with both Python and C++ programming languages. The package generates object hierarchies from detection results by using bounding box overlap to identify parent-child relationships. Experimental results show that the proposed method accurately extracts object hierarchies from images, providing a practical tool to enhance object detection capabilities. The source code for this approach is available at https://github.com/saravit-soeng/HiExtract.**

***Keywords*—Object hierarchy; object relationship; object detection; computer vision**

I. Introduction

Computer vision, particularly object detection and hierarchy extraction in digital images, plays a crucial role in enabling computers to perceive and understand digital images similarly to humans. These technologies facilitate a range of tasks, such as image classification, object detection, and instance segmentation, by allowing the classification of images, identification of objects within images, and segmentation of objects from images [1], [2]. These tasks have been successfully accomplished using advanced deep learning techniques, notably Convolutional Neural Networks (CNNs). By leveraging big data and powerful computational resources, CNNs have significantly enhanced prediction performance and accuracy [3], elevating the field of computer vision. In the current digital era, computer vision tasks have uncovered applications across a broad spectrum of research areas [2], [4].

Object detection is a computer vision task that deals with identifying instances of objects such as humans, cars, motorcycles, or animals in digital images [5], [6], [7]. The results of object detection can be applied in various contexts and fields, including facial recognition, medical image analysis, surveillance systems, robotics, and autonomous driving. Consequently, object detection is one of the most ubiquitous tasks in these diverse applications [6], [7], emphasizing its critical importance in real-world scenarios.

However, obtaining an object hierarchy from images or videos is a challenging task without the assistance of advanced techniques like machine learning or deep learning. Existing object detection methods primarily focus on identifying individual objects within an image and do not establish hierarchical relationships. For example, algorithms such as YOLO (You Only Look Once), SSD (Single Shot MultiBox Detector), Fast R-CNN, and Faster R-CNN offer state-of-the-art approaches for identifying various objects in digital images or videos but do not provide a hierarchical structure that elucidates relationships between objects. This gap in existing methodologies highlights the need for a novel approach to extract hierarchical relationships among detected objects.

This study aims to extract object hierarchies by leveraging the unique combination of object detection results and hierarchical structuring techniques. The task of object hierarchy extraction focuses on establishing parent-child relationships between identified objects within digital images. To construct the object hierarchy, a simple method based on the criteria of overlapping bounding boxes obtained from object detection results is employed. Objects with overlapping bounding boxes are considered to have a parent-child relationship. Thus, object detection serves as a fundamental component of the proposed approach. The study implements the proposed approach as a standalone package compatible with both Python and C++ programming languages, facilitating easy integration into various applications.

This research significantly contributes to the field of computer vision by providing a novel and efficient method for extracting object hierarchies using object detection results. The approach is practical due to its implementation as a standalone package compatible with Python and C++. The versatility of this approach allows it to be integrated into a wide range of applications. The significance of this research lies in its potential to offer new possibilities for understanding and processing digital images in a more structured and relational manner. This work not only extends object detection technology but also opens up possibilities for more complex and nuanced computer vision tasks, marking an important advancement in the field of computer vision.

This paper is organized as follows: Section II reviews related work in object detection and visual relationship extraction. Section III details our proposed method for hierarchical structuring of detected objects. Section IV presents

[*] Corresponding author.

This work was supported by the research grant of the Chungbuk National University in 2024.

experimental results demonstrating the effectiveness of our approach across various scenarios. Section V explores practical applications and use cases for the extracted object hierarchies. Section VI provides insights, discusses limitations, and offers future research directions. Finally, Section VII concludes the paper.

II. Related Work

Object detection and visual relationship extraction have been active areas of research in computer vision. This section provides an overview of recent advancements in these fields and identifies the gap that our research aims to address.

A. Object Detection

Recent years have seen significant progress in object detection algorithms, with several state-of-the-art methods emerging:

1) YOLO (You Only Look Once): Introduced by Redmon et al. [8], YOLO revolutionized object detection by treating it as a regression problem, enabling real-time detection with high accuracy.

2) SSD (Single Shot MultiBox Detector): Liu et al. [9] proposed SSD, which improved upon YOLO by using multi-scale feature maps for detection, enhancing accuracy for objects of various sizes.

3) Fast R-CNN and Faster R-CNN: Girshick [10] and Ren et al. [11] developed these region-based convolutional network methods, which significantly improved detection speed and accuracy.

While these algorithms excel at identifying individual objects, they do not establish hierarchical relationships between detected objects, which is the focus of our research.

B. Visual Relationship Detection

To bridge this gap, several novel frameworks have been proposed to detect visual relationships between objects. Dai et al. [12] introduced a deep relational network that leverages statistical dependencies between objects to detect visual relationships, demonstrating superior performance on two large datasets compared to other state-of-the-art methods. Kolesnikov et al. [13] proposed a model utilizing a Box Attention mechanism to detect visual relationships such as ”person riding motorcycle” or ”bottle on table,” enabling the modeling of pairwise interactions between objects using standard object detection pipelines.

Lu et al. [14] incorporated language priors from semantic word embedding to guide model predictions, training separate models for objects and predicates independently before combining them to predict multiple relationships per image. Zhu et al. [15] introduced deep structured learning for visual relationship detection, employing both feature-level and label-level predictions to learn relationships, thereby capturing dependencies between objects and predicates and enhancing the understanding of visual relationships.

Additional research [16], [17], [18], [19], [20], has contributed various approaches for extracting visual relationships between objects in digital images and videos, achieving notable results. These studies typically formulate visual relationships as <subject-predicate-object> structures.

C. Research Gap and Our Contribution

While existing research has made significant strides in object detection and visual relationship extraction, there remains a gap in efficiently extracting hierarchical relationships among objects in images. Most current methods focus on identifying individual objects or pairwise relationships, but do not provide a comprehensive hierarchical structure.

Our research addresses this gap by proposing a simple yet effective method for extracting object hierarchies from images based on object detection results. Unlike previous work that focuses on complex visual relationships, our approach simplifies the task by establishing a parent-child hierarchy among objects. This method provides a novel perspective on object detection and structuring in computer vision, with potential applications in scene understanding, image captioning, and visual question answering.

By leveraging the output of existing object detection algorithms and introducing a hierarchical structuring technique, our approach offers a practical solution that can be easily integrated into various computer vision applications. This research thus bridges the gap between individual object detection and complex visual relationship extraction, providing a middle ground that captures essential object hierarchies in a computationally efficient manner.

III. Hierarchical Structuring Method of Detected Objects

The proposed research provides a novel method for efficiently creating a hierarchy of detected objects in a digital image. This method simplifies the process of understanding object relationships and interactions, improving accuracy and reducing computational resources. It utilizes a unique combination of object detection algorithms and a hierarchical structuring technique that distinguishes it from existing methods. The proposed method leverages the output of an object detection algorithm, which identifies and locates objects within an image. The object detection results provide bounding boxes and confidence scores for each detected object. These bounding boxes represent the spatial locations of the objects, while the confidence scores indicate the likelihood that the detected objects are actually present in the scene [8].

The proposed method for extracting object hierarchies from object detection results has been implemented as a versatile, cross-platform package, facilitating seamless integration into diverse applications across multiple domains. The package is available for both Python and C++ programming languages, ensuring wide accessibility and compatibility with the preferred development environments of a vast range of developers. To leverage the capabilities of this package, users simply need to provide the object detection results as input, typically comprising predicted object classes and their corresponding bounding box coordinates. The package then processes this input data, employing sophisticated algorithms to analyze the spatial relationships between the detected objects and construct

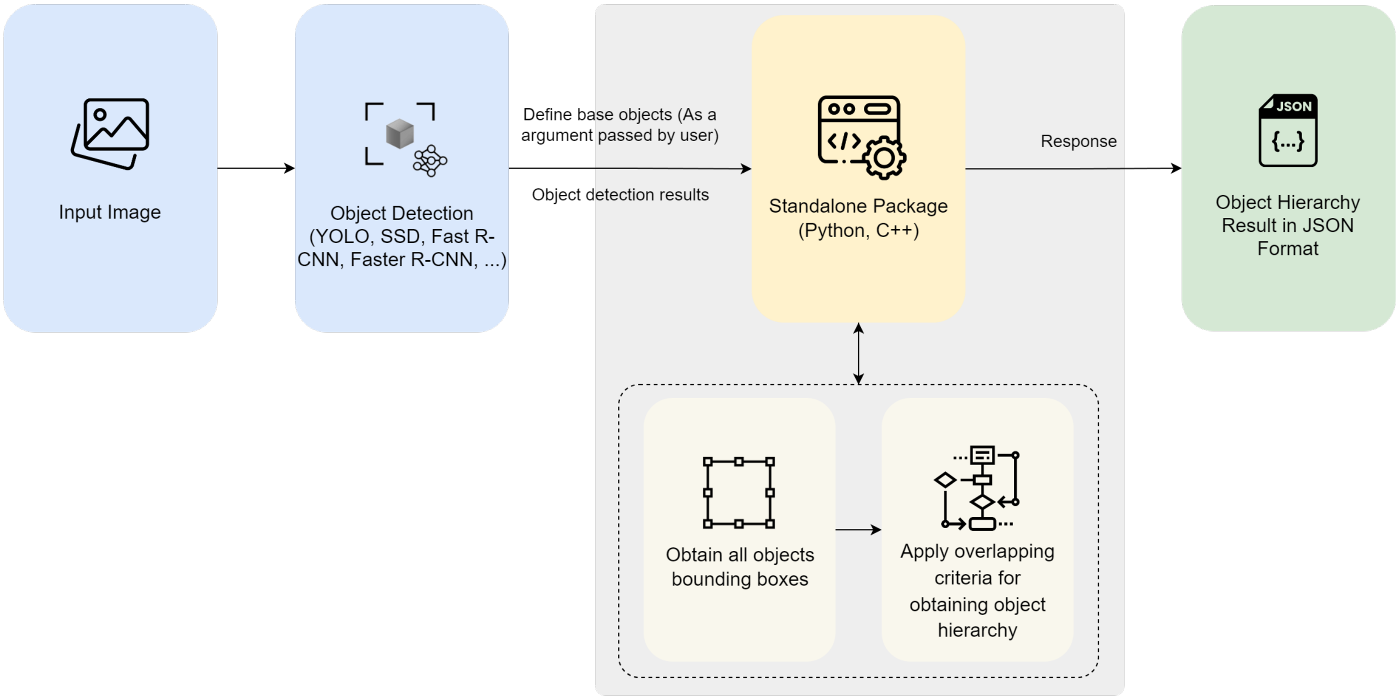

Input Image
Object Detection (YOLO, SSD, Fast R-CNN, Faster R-CNN, ...)
Define base objects (As a argument passed by user)
Object detection results
Standalone Package (Python, C++)
Response
JSON
{...}
Object Hierarchy Result in JSON Format
Obtain all objects bounding boxes
Apply overlapping criteria for obtaining object hierarchy

Fig. 1. A flowchart illustrating the process of object detection and hierarchy establishment.

Algorithm 1 Extract the object hierarchy from digital image based on object detection

```
function EXTRACT_OBJECT_HIERARCHY(base_objects, result)
    get predicted classes and bounding boxes from detection result
    combine predicted classes and bounding boxes: object_with_boxes
    get all class names
    initial empty result list: result_list []
    for each object in base_objects do
        get base object class index from original classes
        if object_with_boxes do not empty then
            for each object in object_with_boxes do
                if class index of current object == class index of base object then
                    create children object: children []
                    for each object in object_with_boxes do
                        if not base object then
                            if current object overlaps with base object then
                                add child object to children list
                            end if
                        end if
                    end for
                    if children object is not empty then
                        create a parent object with child objects
                        add parent object to result_list
                    end if
                end if
            end for
        end if
    end for
    create results object with result_list data
    return results object
end function
```

a hierarchical representation of the scene. The object hierarchy extraction process is illustrated in Fig. 1.

The given pseudo-code in Algorithm 1 describes a function called extract_object_hierarchy that takes two inputs: base_objects and result. The core algorithm accepts only two primary inputs:

1) Base_objects: This parameter specifies the parent objects from which we aim to extract the hierarchy. It allows flexibility in defining the root nodes of our hierarchical structure.

2) Result: This parameter encompasses the complete object detection results, typically including bounding box coordinates and class predictions for all detected objects in the image.

The purpose of this function is to analyze the output of an object detection algorithm and organize the detected objects into a hierarchical structure based on their spatial relationships (overlapping) and their predicted classes.

Initially, the package combines the predicted object classes and their associated bounding boxes into a unified data structure. The package then iterates over each detected object, referred to as the ”base object,” retrieving its corresponding class index from the original set of classes. If the detection results contain overlapping objects, the package performs an additional loop to identify objects whose class indices match that of the current base object. For each matching object, the package creates a new list to store objects that spatially overlap with the base object. It iterates over the detection results once more, comparing the bounding boxes of the objects against the base object's bounding box. If an overlap is detected, the current object is appended to the ‘children‘ list.

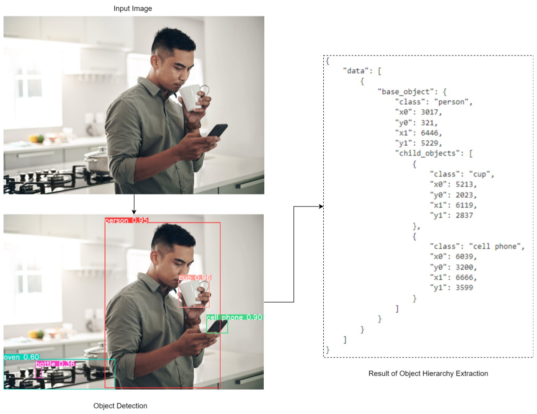
Upon completing the overlap detection process, if the ‘children‘ list is not empty, a new ”parent object” is constructed, encapsulating the base object and its associated children objects. This parent object is then added to the ‘result_list‘, effectively building the hierarchical structure.

After processing all base objects and their corresponding overlapping objects, the package consolidates the ‘result_list‘ data into a final results object, which is then returned to the user. This results object represents the complete hierarchical structure of the detected objects, enabling users to seamlessly integrate and leverage this information within their applications.

IV. Experimental Results

The experiments were conducted on various images in different scenarios to evaluate the proposed approach's effectiveness in extracting object hierarchies. The proposed method demonstrated impressive performance in extracting object hierarchies from images. However, the accuracy of the object hierarchy extraction process heavily relies on the object detection results.

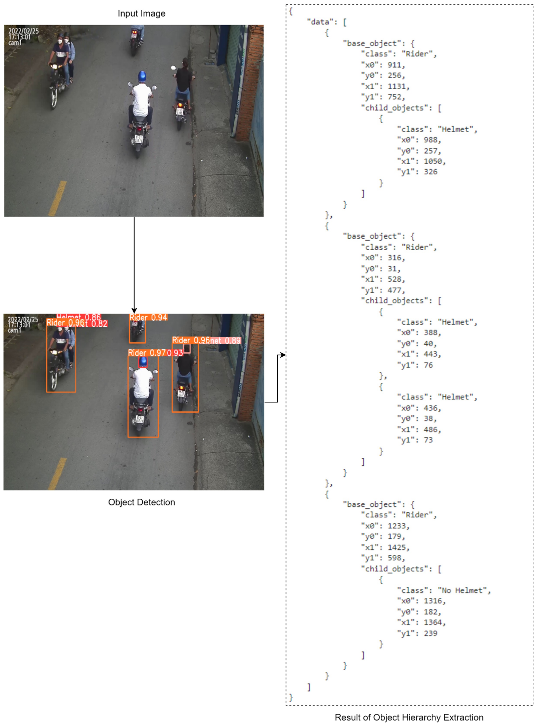
The first experiment involved an image of a person holding a cell phone and a cup. Based on the proposed method, the person was defined as the single base or parent object, while the cell phone and cup were detected as child objects. This experiment utilized the YOLOv8s [21] model for object detection from images, as illustrated in Fig. 2.



```
{
    "data": [
        {
            "base_object": {
                "class": "person",
                "x0": 3017,
                "y0": 321,
                "x1": 6446,
                "y1": 5229,
                "child_objects": [
                    {
                        "class": "cup",
                        "x0": 5213,
                        "y0": 2023,
                        "x1": 6119,
                        "y1": 2837
                    },
                    {
                        "class": "cell phone",
                        "x0": 6039,
                        "y0": 3200,
                        "x1": 6666,
                        "y1": 3599
                    }
                ]
            }
        }
    ]
}
```

Fig. 2. An experimental result on single base object.

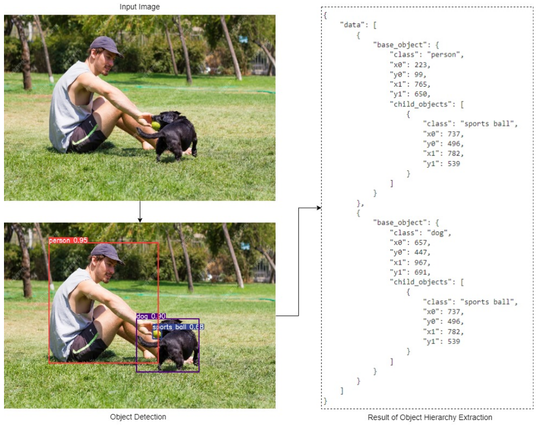
Another experiment was conducted on an image containing multiple common base objects. In this case, a custom YOLOv5 model was employed to detect objects from the image. The custom model identified three distinct objects: rider, helmet, and non-helmet. The rider was defined as the base object, and through the extraction process, multiple parent-child relationships were established between the rider object and the helmet or non-helmet objects. Fig. 3 illustrates the results obtained in this experiment.



```
{
    "data": [
        {
            "base_object": {
                "class": "Rider",
                "x0": 911,
                "y0": 256,
                "x1": 1131,
                "y1": 752,
                "child_objects": [
                    {
                        "class": "Helmet",
                        "x0": 988,
                        "y0": 257,
                        "x1": 1050,
                        "y1": 326
                    }
                ]
            }
        },
        {
            "base_object": {
                "class": "Rider",
                "x0": 316,
                "y0": 31,
                "x1": 528,
                "y1": 477,
                "child_objects": [
                    {
                        "class": "Helmet",
                        "x0": 388,
                        "y0": 40,
                        "x1": 443,
                        "y1": 76
                    },
                    {
                        "class": "Helmet",
                        "x0": 436,
                        "y0": 38,
                        "x1": 486,
                        "y1": 73
                    }
                ]
            }
        },
        {
            "base_object": {
                "class": "Rider",
                "x0": 1233,
                "y0": 179,
                "x1": 1425,
                "y1": 598,
                "child_objects": [
                    {
                        "class": "No Helmet",
                        "x0": 1316,
                        "y0": 182,
                        "x1": 1364,
                        "y1": 239
                    }
                ]
            }
        }
    ]
}
```

Fig. 3. An experimental result on multiple common base objects.

In addition to defining a common base object, the proposed approach was tested on multiple base objects. Fig. 4 shows the results obtained from an image featuring a person, a dog, and a tennis ball. In this test, both the person and the dog were defined as base objects.



```
{
    "data": [
        {
            "base_object": {
                "class": "person",
                "x0": 223,
                "y0": 99,
                "x1": 765,
                "y1": 650,
                "child_objects": [
                    {
                        "class": "sports ball",
                        "x0": 737,
                        "y0": 496,
                        "x1": 782,
                        "y1": 539
                    }
                ]
            }
        },
        {
            "base_object": {
                "class": "dog",
                "x0": 657,
                "y0": 447,
                "x1": 967,
                "y1": 691,
                "child_objects": [
                    {
                        "class": "sports ball",
                        "x0": 737,
                        "y0": 496,
                        "x1": 782,
                        "y1": 539
                    }
                ]
            }
        }
    ]
}
```

Fig. 4. An experimental result on different base objects.

The proposed approach demonstrated its versatility by handling various object hierarchies, ranging from a single base object to multiple base objects coexisting within an image. However, it is crucial to acknowledge that the accuracy of the extracted object hierarchy heavily depends on the performance of the underlying object detection model. Improvements in object detection algorithms could further enhance the accuracy and robustness of the proposed object hierarchy extraction method.

V. Application Use Cases

The extracted object hierarchy results can be leveraged in a variety of research and practical applications. In this research, we explored two distinct use cases to demonstrate the utility of the proposed object hierarchy extraction method.

The first use case involved the detection of motorcycle riders without helmets, a critical safety concern. We employed a custom YOLOv5-based model to detect three distinct objects: rider, helmet, and non-helmet. Leveraging the proposed method, a standalone package was developed to generate object hierarchy results from the object detection outputs. These object hierarchy results enabled the identification of illegal actions by motorcycle riders, specifically those not wearing helmets. Fig. 5 illustrates an example of detecting motorcycle riders without helmets using the generated object hierarchy results.

In this application, the object hierarchy played a crucial role in distinguishing between riders wearing helmets and those without helmets. By establishing the parent-child relationships between the rider object and the helmet or non-helmet objects, the system could effectively identify instances where a rider was associated with a non-helmet object, indicating a potential safety violation.

The second use case explored the application of object hierarchy results in a different domain or scenario, leveraging the versatility of the proposed approach. In this use case, the object hierarchy results enable the determination of whether specific items belong to a particular person within a defined area. By establishing the parent-child relationships between the person object and the item objects (e.g., handbag, backpack, or any other prohibited item), the system can effectively identify instances where an individual is carrying a restricted item into a prohibited area.

Fig. 6 demonstrates an example of detecting a person carrying a handbag into a banned area based on the proposed method. The object hierarchy results allow the system to associate the handbag object with the person object, indicating that the individual is in possession of the item while entering the restricted area.

The versatility of the proposed object hierarchy extraction method allows for its application in diverse domains, showcasing its potential for enhancing situational awareness, decision-making processes, and automated analysis of complex visual data.

VI. Discussion

The proposed method for extracting object hierarchies from images based on object detection results offers a novel approach to understanding spatial relationships between objects. This section discusses the implications of our findings, limitations of the current study, and potential directions for future research.

A. Implications of the Findings

Our approach demonstrates the feasibility of extracting meaningful hierarchical relationships between objects in images using a straightforward method based on bounding box overlap. This finding carries several important implications:

1) Simplified scene understanding: By organizing detected objects into a hierarchical structure, our method provides a more intuitive representation of the scene, potentially simplifying downstream tasks such as image captioning or visual question answering.

2) Computational efficiency: Compared to more complex visual relationship detection methods, our approach is computationally efficient, making it suitable for real-time applications.

3) Versatility: As demonstrated in our use cases, the extracted hierarchies can be applied to various domains, from traffic safety to security applications.

B. Limitations of the Study

While our method demonstrates potential, it is important to acknowledge its limitations:

1) Dependence on object detection accuracy: The quality of the extracted hierarchy is heavily dependent on the accuracy of the underlying object detection algorithm. Errors in object detection can propagate to the hierarchy extraction process.

2) Simplistic relationship model: Our method primarily relies on spatial overlap to determine relationships, which may not capture more complex or abstract relationships between objects.

3) Lack of semantic understanding: The method does not incorporate semantic knowledge about object classes, which could potentially improve the accuracy of the extracted hierarchies.

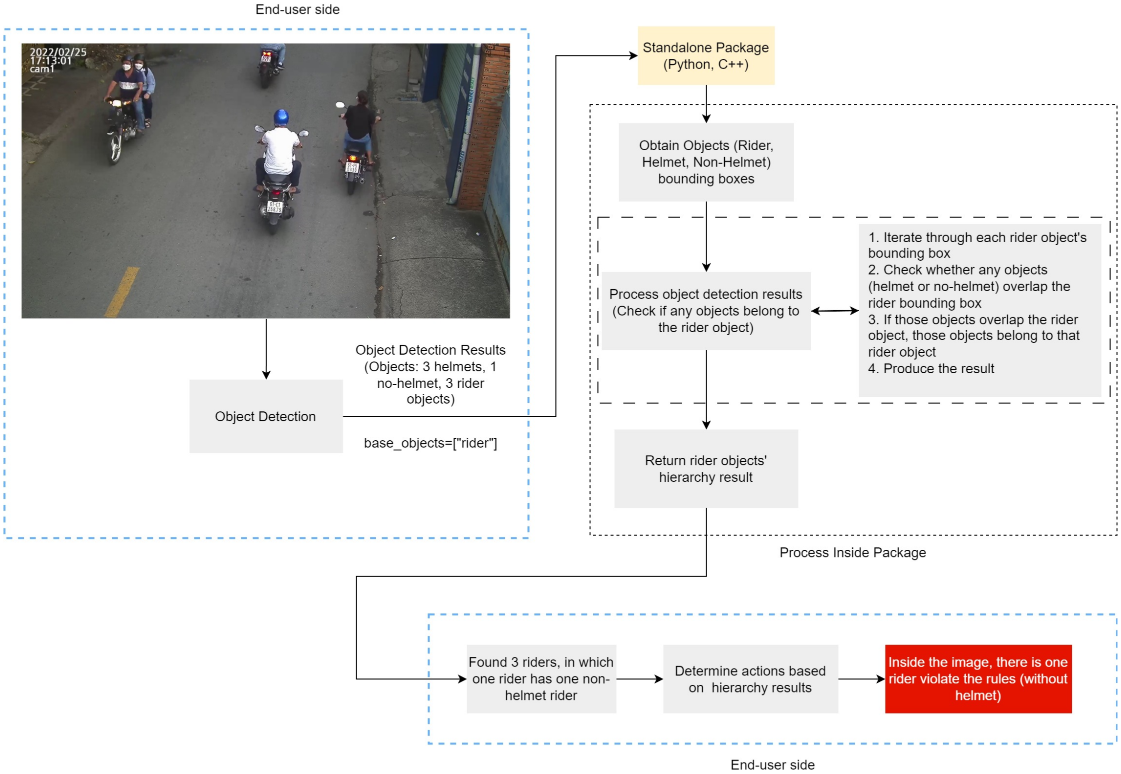

End-user side
2022/02/25
17:13:01
cam1
Object Detection
Object Detection Results (Objects: 3 helmets, 1 no-helmet, 3 rider objects)
base_objects=["rider"]
Standalone Package (Python, C++)
Obtain Objects (Rider, Helmet, Non-Helmet) bounding boxes
Process object detection results (Check if any objects belong to the rider object)
1. Iterate through each rider object's bounding box
2. Check whether any objects (helmet or no-helmet) overlap the rider bounding box
3. If those objects overlap the rider object, those objects belong to that rider object
4. Produce the result
Return rider objects' hierarchy result
Process Inside Package
Found 3 riders, in which one rider has one non-helmet rider
Determine actions based on hierarchy results
Inside the image, there is one rider violate the rules (without helmet)
End-user side

Fig. 5. An example of detecting motorcycle riders without wearing helmets using the object hierarchy results.

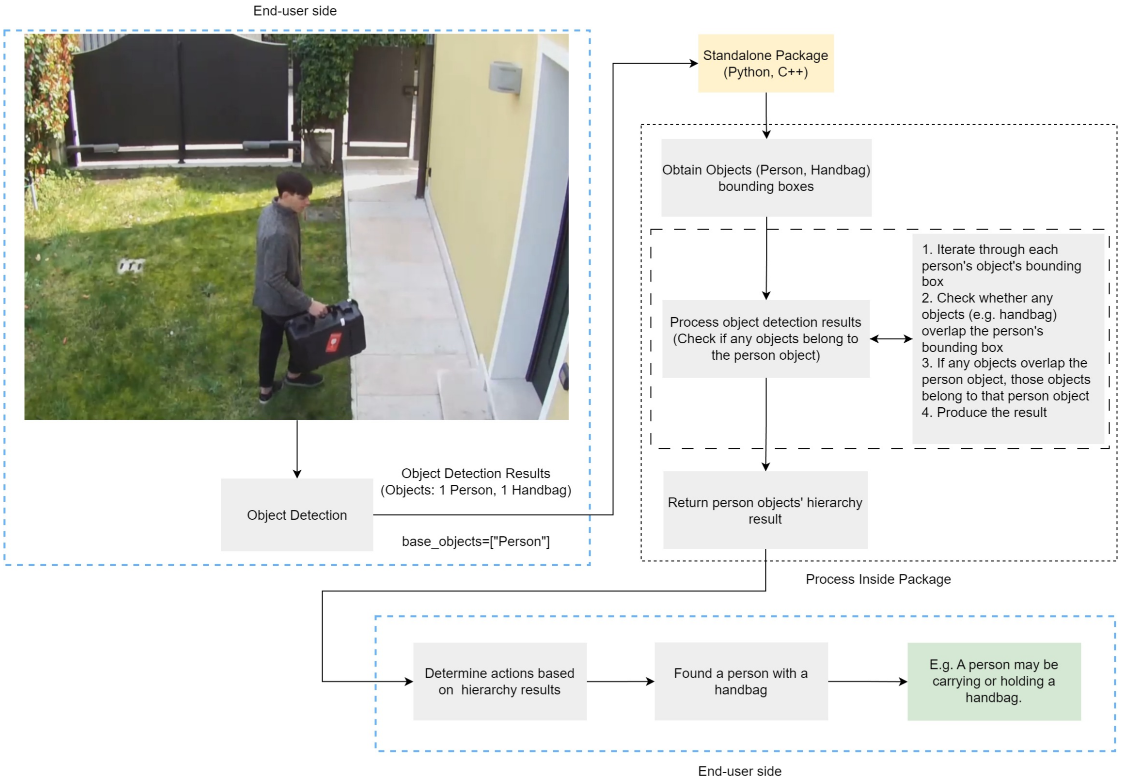

End-user side
Object Detection
Object Detection Results (Objects: 1 Person, 1 Handbag)
base_objects=["Person"]
Standalone Package (Python, C++)
Obtain Objects (Person, Handbag) bounding boxes
Process object detection results (Check if any objects belong to the person object)
1. Iterate through each person's object's bounding box
2. Check whether any objects (e.g. handbag) overlap the person's bounding box
3. If any objects overlap the person object, those objects belong to that person object
4. Produce the result
Return person objects' hierarchy result
Process Inside Package
Determine actions based on hierarchy results
Found a person with a handbag
E.g. A person may be carrying or holding a handbag.
End-user side

Fig. 6. An example of detecting a person carrying a handbag into a banned area based on the proposed method.

C. Future Research Directions

Based on the findings and limitations of this study, several avenues for future research emerge:

1) Integration with semantic knowledge: Incorporating class-specific semantic information could enhance the accuracy of the hierarchy extraction process.

2) Temporal hierarchies: Extending the approach to video data to capture temporal relationships between objects over time.

3) Machine learning enhancement: Developing a machine learning model to predict hierarchical relationships based on both spatial and semantic features could improve the method's accuracy and robustness.

4) Expanding object detection compatibility: Future versions of the standalone package should support a wider range of object detection algorithms beyond YOLO.

While our proposed method offers a simple and efficient approach to extracting object hierarchies, there is significant potential for further refinement and expansion of this technique. Future research should focus on addressing the current limitations and exploring more sophisticated approaches to understanding object relationships in visual data.

VII. Conclusion

In conclusion, this research has significantly contributed to the field of computer vision by introducing a novel approach for extracting object hierarchies from digital images. Utilizing the results of object detection, the proposed research presents a unique method that identifies parent-child relationships between objects based on overlapping bounding boxes criteria. This approach is also practical, as it has been implemented as a standalone package compatible with both Python and C++ programming languages. The versatility of this approach allows it to be integrated into a wide range of applications. The implications of this research are solid, offering new possibilities for understanding and processing digital images in a more structured and relational manner. This work not only expands object detection techniques but also opens up possibilities for more complex and subtle computer vision tasks, marking a significant step forward in the field. Based on the experiments, the results have demonstrated the effectiveness of the proposed approach in extracting the object hierarchy from the images. However, in terms of standalone package implementation, object detection is limited to the YOLO algorithms. In the next release version, we intend to expand to other object detection algorithms and improve the approach to work on more complexities of hierarchy extraction.

AUTHORS' CONTRIBUTION

Saravit Soeng: Conceptualization; methodology; coding; writing original draft. Vungsovanreach Kong: Conceptualization; validation; writing review and editing. Munirot Thon: Conceptualization; writing review and editing. Wan-Sup Cho: Investigation; resources. Tae-Kyung Kim: Conceptualization; methodology; supervision; project administration.

Acknowledgments

This research was funded by PentaGate, a company based in Republic of Korea.

Conflict of interest

The authors declare no potential conflict of interests.

Data Availability Statement

The images used in this study were obtained from publicly available sources. No new images were generated as part of this research.

References

[1] A. Esteva, K. Chou, S. Yeung, N. Naik, A. Madani, A. Mottaghi, Y. Liu, E. Topol, J. Dean, and R. Socher, "Deep learning-enabled medical computer vision," *NPJ digital medicine*, vol. 4, no. 1, p. 5, 2021.

[2] D. Bhatt, C. Patel, H. Talsania, J. Patel, R. Vaghela, S. Pandya, K. Modi, and H. Ghayvat, "Cnn variants for computer vision: History, architecture, application, challenges and future scope," *Electronics*, vol. 10, no. 20, p. 2470, 2021.

[3] N. O'Mahony, S. Campbell, A. Carvalho, S. Harapanahalli, G. V. Hernandez, L. Krpalkova, D. Riordan, and J. Walsh, "Deep learning vs. traditional computer vision," in *Advances in Computer Vision: Proceedings of the 2019 Computer Vision Conference (CVC), Volume 1 1*. Springer, 2020, pp. 128–144.

[4] S. Paneru and I. Jeelani, "Computer vision applications in construction: Current state, opportunities & challenges," *Automation in Construction*, vol. 132, p. 103940, 2021.

[5] Z. Zou, K. Chen, Z. Shi, Y. Guo, and J. Ye, "Object detection in 20 years: A survey," *Proceedings of the IEEE*, vol. 111, no. 3, pp. 257–276, 2023.

[6] Y. Amit, P. Felzenszwalb, and R. Girshick, *Object Detection*. Cham: Springer International Publishing, 2021, pp. 875–883. [Online]. Available: $https://doi.org/10.1007/978-3-030-63416-2_{6}60$

[7] Z.-Q. Zhao, P. Zheng, S.-t. Xu, and X. Wu, "Object detection with deep learning: A review," *IEEE transactions on neural networks and learning systems*, vol. 30, no. 11, pp. 3212–3232, 2019.

[8] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real−time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[9] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*. Springer, 2016, pp. 21–37.

[10] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1440–1448.

[11] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 6, pp. 1137–1149, 2016.

[12] B. Dai, Y. Zhang, and D. Lin, "Detecting visual relationships with deep relational networks," in *Proceedings of the IEEE conference on computer vision and Pattern recognition*, 2017, pp. 3076–3086.

[13] A. Kolesnikov, A. Kuznetsova, C. Lampert, and V. Ferrari, "Detecting visual relationships using box attention," in *Proceedings of the IEEE/CVF international conference on computer vision workshops*, 2019, pp. 0–0.

[14] C. Lu, R. Krishna, M. Bernstein, and L. Fei-Fei, "Visual relationship detection with language priors," in *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*. Springer, 2016, pp. 852–869.

[15] Y. Zhu and S. Jiang, "Deep structured learning for visual relationship detection," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018.

[16] K. Liang, Y. Guo, H. Chang, and X. Chen, "Visual relationship detection with deep structural ranking," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018.

[17] X. Shang, T. Ren, J. Guo, H. Zhang, and T.-S. Chua, "Video visual relation detection," in *Proceedings of the 25th ACM international conference on Multimedia*, 2017, pp. 1300–1308.

[18] Y. Zhan, J. Yu, T. Yu, and D. Tao, "On exploring undetermined relationships for visual relationship detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 5128–5137.

[19] J. Zhang, Y. Kalantidis, M. Rohrbach, M. Paluri, A. Elgammal, and M. Elhoseiny, "Large-scale visual relationship understanding," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 33, no. 01, 2019, pp. 9185–9194.

[20] B. Zhuang, L. Liu, C. Shen, and I. Reid, "Towards context-aware interaction recognition for visual relationship detection," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 589–598.

[21] G. Jocher, A. Chaurasia, and J. Qiu, "Ultralytics YOLO," Jan. 2023. [Online]. Available: https://github.com/ultralytics/ultralytics