# AN INTEROPERABLE FRAMEWORK FOR PLANETARY DEFENSE DATA INTEGRATION AND VISUALIZATION TO SUPPORT THE MITIGATION OF POTENTIAL HAZARDOUS ASTEROIDS

by

Ishan Shams
A Dissertation
Submitted to the
Graduate Faculty
of
George Mason University
in Partial Fulfillment of
The Requirements for the Degree
of
Doctor of Philosophy
Earth Systems and Geoinformation Sciences

Committee:

| | |
|---|---|
| ________________________ | Dr. Chaowei Yang, Dissertation Director |
| ________________________ | Dr. Ruixin Yang, Committee Member |
| ________________________ | Dr. Andreas Zufle, Committee Member |
| ________________________ | Dr. Wenying Ji, Committee Member |
| ________________________ | Dr. Dieter Pfoser, Department Chairperson |
| ________________________ | Dr. Donna M. Fox,<br>Associate Dean Office of Student Affairs<br>& Special Programs, College of Science |
| ________________________ | Dr. Fernando R. Miralles-Wilhelm<br>Dean, College of Science |
| Date: 04/21/2022 | Spring Semester 2022<br>George Mason University<br>Fairfax, VA |

# An Interoperable Framework for Planetary Defense Data Integration and Visualization to Support the Mitigation of Potential Hazardous Asteroids

A Dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy at George Mason University

by

Ishan A. Shams
Bachelor of Science
George Mason University, 2016

Director: Chaowei Yang, Professor
Department of Geography and Geoinformation Science

Spring Semester 2022
George Mason University
Fairfax, VA

## DEDICATION

This dissertation is dedicated to my father, who has never lost faith in me and always believed that I would be successful. Thank you, dad!

# ACKNOWLEDGMENTS

I would like to express my gratitude for the many mentors, colleagues, friends, and family who have been so integral in helping me achieve today's successes. My mentor and advisor, Dr. Chaowei Yang, was always available to help stimulate creative solutions, which allowed me to be highly successful. Dr. Yang helped develop the research path that led me to where I am now—a scientist with vision and passion pursuing a lifelong dream of making a difference in the world we all share.

I am also thankful for the support I received from Dr. Andreas Zufle, Dr. Wenying Ji, and Dr. Ruixin Yang, who shared their incredible expertise with me during my research at George Mason University. Their support helped point the way forward during those moments of confusion when all seemed lost. My parents and extended family also played an invaluable role by providing love and support during my long absence away from them. They helped make this possible. They are my inspiration for always trying harder and never giving up. Finally, I thank God, for being my foundation and guiding light throughout this journey of life.

Many thanks are also owed to my co-researchers at NSF Spatiotemporal Innovation Center for the great job they did as my team members during the research. It was a pleasure to work with all of them. In particular, I want to acknowledge the efforts of Jingchao, Mengchao, Zifu, Qian, Manzhu, Fei, Yongyao, Lara, Nick, Dexuan, Kyla, Yun, and Kai. I might have missed a few names here. Please forgive me if I have done so—there are far too many to list.

## TABLE OF CONTENTS

## LIST OF TABLES

## LIST OF FIGURES

**LIST OF EQUATIONS**

**ABSTRACT**

AN INTEROPERABLE FRAMEWORK FOR PLANETARY DEFENSE DATA INTEGRATION AND VISUALIZATION TO SUPPORT THE MITIGATION OF POTENTIAL HAZARDOUS ASTEROIDS

Ishan A. Shams, Ph.D.

George Mason University, 2022

Dissertation Director: Dr. Chaowei Yang

A large asteroid impact can cause catastrophic environmental effects, as was shown by the Chicxulub impact some 66 million years ago (Pope et al. 1997). In order to protect our planet from future near-Earth objects (NEOs), it is crucial to efficiently and seamlessly integrate data, discoveries, and resources. However, planetary defense information remains scattered throughout multiple branches, organizations, and countries.

The challenges that come with dispersed planetary defense information are manifold. First, the heterogeneity of planetary defense situations requires unique responses from various organizations. Second, there is a lack of structured integration, and interoperability among planetary defense stakeholders. This hampers effective communication and collaboration. Third, the diversity of data and information for planetary defense research creates discrepancy between PD data formats for individual researchers. Finally, future threats mitigation efforts are often hindered by a lack of

comprehensive understanding of the problem. Consequently, an interoperable framework for planetary defense data integration and visualization is needed to support the mitigation of potentially hazardous asteroids.

This dissertation has presented a data-fusion framework that can be used to support the detection, characterization, and mitigation of potentially hazardous asteroids. The data-fusion framework was used to develop the Planetary Defense Knowledge Gateway (PDKG), a platform that enables users to access, visualize, and analyze integrated, and interoperable planetary defense data. This dissertation also focused on multiprocessing techniques, comprehensive data modeling, and data inaccuracies verification. The implemented multiprocessing techniques provides three main advantages: (1) a data pre-fetching technique to minimize data retrieval latency, (2) an in-memory caching technique to improve data access performance, and (3) a query parallelization technique to speed up the execution of complex queries. The comprehensive data modeling considered the different types of information that needed to be integrated, such as observational data, catalog data, and expert knowledge. The data inaccuracies verification was performed using a set of heuristics that were designed to identify errors in the data.

This research provides a foundation upon which the planetary defense community can build to mitigate the effects of dispersed information and aid in the overall decision-making strategies.

# 1. INTRODUCTION

Near-Earth objects (NEOs) are asteroids, comets, and large meteoroids whose orbit intersects with the Earth's orbit; as such, they may pose a collision danger. NEOs are composed of mostly water ice with embedded dust particles. Many physical characteristics, such as albedo, brightness, shape, and phase, can be defined and measured for NEOs, particularly in relation to asteroids and comets. The scientific interest in comets and asteroids is mainly due to the possibility that they may collide with our planet, which represents a hazard to life on Earth (Farnocchia et al. 2013). An asteroid has the potential to cause havoc to the global climate if it collides with Earth. In addition, asteroids can also impact large areas of land or water near their point of contact with Earth, which can damage buildings, animals, vegetation, and human populations. Some 66 Ma ago, a large impactor between 10 to 90 kilometers in diameter struck Chicxulub, Mexico (Pope et al. 1997) with catastrophic environmental effects. According to the report (Pope et al. 1997), the impactor's shockwave deposited large amounts of rock and soil on the planet's atmosphere, causing a global climate change, which led to the extinction of dinosaurs and most of the planet's large animals (Napier 2015). Although such events are frequent in nature, a more recent incident, specifically, the Chelyabinsk impact in Russia on February 15, 2013, highlighted the importance of protecting our planet from future near-Earth objects (NEO) (Popova et al. 2013).

Significantly, as many scientific studies show, the impacts of NEOs have contributed to mass extinctions and evolution (Chiarenza et al. 2020; Pope et al. 1994; Rampino, Caldeira, and Prokoph 2019). Moreover, research demonstrates that NEOs will continue to hit Earth at irregular intervals in the future (Perna et al. 2015). Impactors range from benign fireballs to large airbursts, the largest of which can cause tragic destruction on the ground. However, these large airbursts are very unlikely to occur in any given lifetime, and they are probably randomly distributed in time. For events involving harmless fireballs, the methods of civil defense are sufficient for saving human lives. For more massive airbursts, changing the path of these near-Earth objects reaching Earth's vicinity is the appropriate solution. Additionally, for global catastrophic events that cause mass extinctions, there is currently no technology that is capable of avoiding disaster.

Global organizations have undertaken dozens of research studies and many scientific explorations to mitigate the potential impact of near-Earth objects. Programs, like NASA's NEO Survey, for instance, assist in educating the public and is an important tool in gaining support for impact mitigation decision-making in the United States (Arentz et al. 2010; Larson 2006). At the same time, numerous NASA-funded astronomer teams are constantly searching for potentially hazardous Near-Earth objects, whose orbits periodically bring them within 30 million miles of the Earth's orbit. At NASA, the Planetary Defense Coordination Office not only supports these search programs, but they also, in line with the United States' *National Near-Earth Object Preparedness Strategy and Action Plan* policies (Daou and Johnson 2019), plan and coordinate responses to prepare for an NEO event. As part of NASA's planetary defense strategy, the Center for Near-Earth Object Studies at Jet Propulsion Laboratory (JPL), analyzes the data collected on near-Earth objects and publishes the data and types discovered (Yeomans et al. 2001; Yeomans, Chesley, and Chodas 2010). Numerous efforts can be undertaken to mitigate the hazard of potential asteroid impacts, including emergency response planning, civil defense, slow-push or pull methods, kinetic impact, deflection mission concept studies, and nuclear detonation. As part of their defense strategy, the European Union's (EU) NEOShield Project is considering kinetic impactor options, the latest deflection techniques, and gravity-tractor methods. At present, the EU’s Department of Energy's national laboratories are responsible for studying the impact effects of these strategies.  Meanwhile, NASA’s Planetary Defense Coordination Office (PDCO) is also collaborating with other US Government agencies, as well as other national and international agencies, to ensure the detection of potentially hazardous objects (Johnson 2016). As a result of this disunity, much of the work of many distinct Planetary Defense (PD) experts around the world is not unified or synchronous.

Currently, information about detecting, characterizing, and mitigating NEO threats is dispersed throughout different multiple branches, organizations, and countries due to the lack of structured architecture. This dispersion can cause errors at the time of crisis, resulting in miscalculated mitigation efforts. There are three relevant aspects for integrating the resources and producing cohesive solutions to PD mitigation: 1) data and information identification, 2) communication and international engagement, and 3) knowledge base.

1) Data/Information Identification: Detecting and processing data about whether an NEO poses a real impact threat, such as the characterization of the PHA's spatiotemporal occurrence, is the first step in any PD effort. The recognition of capabilities within different organizations for data processing and simulation is the pinnacle in any mitigation plan. The objective is to identify information that can be connected to support coordinated observations, verify observations (or extensions of physics-based models), and produce appropriate mitigation plans.
2) Communication and International Engagement: The ability to leverage international capacities for observation, data processing, simulation development, knowledge sharing, and actionable solutions is critical in threat mitigation. Some PD-related missions have incorporated international collaboration, notably OSIRIS-Rex (Garner 2015) and DART (Talbert 2017). To avoid the risks caused by different countries implementing their own mitigation strategies to the same hazard, it's critical to establish a communication mechanism among nations with mitigation capacity.
3) Knowledge Base: The percentage of a PHA striking Earth within the next five to ten years has decreased over time owing to continuous research into large space rocks. However, the next 100 to 300 years is less predictable. With efficiency and a higher probability of success, the research presented in this dissertation suggests the formation of a data/knowledge base that combines research findings and expert knowledge to confront future PHAs with speed and efficacy (Shams et al. 2019).

Knowledge discovery is essential to the overall success of how research is conducted in the Planetary Defense industry. Despite its importance, knowledge discovery has been explored heavily due to its highly complex implementation. There are several steps that are needed to be taken, which include understanding the PD domain, available data sources, and preparation of data. Once the industry is equipped with a multinational database that integrates data from agencies, such as NASA, ESA, JAXA, Minor Planet Center, and other authentic open-source communities, then this allows the knowledge discovery to be highly valuable. A multinational database that integrates data from the stated sources is a challenge far beyond the scope of this research, as it requires authorization and acceptance of all agencies to work together. In this defense, the scope was reduced to build a proof of concept that integrates a few selected data sources that are open to the public. These data sources are mentioned in Chapter 3. Several experiments have been conducted with this proof-of-concept, data-fused database that has shown signs of several benefits.

Several whitepapers in the western countries have outlined the challenges that are related to the problem of unorganized and scattered data. These challenges affect how research is conducted, resulting in either incomplete data or calculations that must incorporate a higher margin of error (Franke and Nielson 1991; Lodha and Franke 1997; Remondino 2003; Yang, Deng, and Chen 2005; Zhao et al. 2000). To address this issue, this dissertation used several data fusion techniques to demonstrate how merged data can produce a bigger picture, improve knowledge discovery, and reduce the level of inaccuracies. A proper data fusion implementation not only provides scientists with a new way to approach a problem, but it also improves the capability of enhanced data discovery. In planetary defense, in general cases, if someone plans to calculate NEO coordinates, they must rely on a specific observatory dataset. Further, if the satellite trajectory is added to the calculations, then this becomes an additional challenge. However, this type of analysis can be done with ease if a data-fused database is utilized.

Fusion techniques are also experiencing a steady growth in the scientific community due to their importance. The way industries implement fusion techniques varies according to the needs of domains. For instance, Jusoh and Almajali (2020) discuss several fusion techniques and approaches developed in the robotics, military, and health care field in the United States. Castanedo (2013) also reviewed several classification data fusion techniques that can be implemented to complement numerous data sources with each other. In a more recent study, Ntumba, Gore, and Awanyo (2021) applied data fusion techniques to predict Apophis Asteroid Flyby's optimal trajectories. Along with data fusion, they also applied a neural network model to track and predict asteroids' orbits. There are several additional studies discussed in Chapter 2.2 that demonstrate data fusion in detail, along with data and information sources, strategies, and its challenges.

Interfaces have long played an essential role in the progress of information science, assisting users in locating required information quickly and accurately from vast data. This has subsequently altered the way people search for and obtain knowledge. In addition to general search engines (e.g., Google, Bing), multiple vertical domain search engines have been developed to give a customizable searching feature for users from specific domains or with similar information goals. Given the volume of search requests, it makes sense to save users time by indexing pages in just one or two domains at a time. The crawling and data storage are linked to a specific domain, so saving time for individuals who want information using the available searching capabilities is beneficial.

Data discovery refers to the process of identifying and extracting meaningful information from data sources for analysis (Curry 2016). The challenge is that data can be scattered across different platforms, which creates a fragmented view of reality. Data analysts need to spend significant amounts of time collecting and compiling all this disparate data into one place to make sense of it. This is where search interfaces come in: They provide a way for users to explore their planetary defense-related data with ease and speed, no matter how dispersed or complex it may be. With a powerful set of tools at hand, these interfaces enable you to find the insights hidden within your data.

The discovery of information regarding how to improve data management and information access in the current, widely dispersed format is one of the primary aims of this dissertation. Search interfaces provide users with different ways to search for relevant documents within a specific domain or across multiple domains. Searching for information in a particular domain can be enabled by vertical-specific crawlers, which make it possible to discover and extract metadata and content related to near-Earth potential hazardous objects. However, crawlers can only traverse publicly accessible web pages and neglect private data sources, such as databases and application program interfaces (APIs), which cannot be accessed through crawling due to privacy reasons. This research discusses how search APIs can provide efficient access to data from different sources with a minimal number of calls to the API compared with accessing an entire website or application program interface (API) service.

One of the problems addressed in this dissertation is to assist researchers who lack information access capabilities. These "information curators" usually work on specific aspects of domain data management within their organization's working context, which restricts them from performing more extensive metadata indexing those cross organizational lines beyond their boundaries.

In the United States, there are currently three ground-based NEO search projects to monitor, track, and find NEOs via visualization. The Planetary Society recognizes the danger that asteroids and comets pose (Pelton 2021), which is why they have been using data visualization technologies for over a century, as a way to convey complicated information in a graphical or pictorial format. Data visualization aids in the understanding of why things occur as well as the comparison of various patterns and trends that might impact future events. These programs must be successful since, although over 1800 potentially hazardous objects have been recorded (size > 140 m), this number is growing every year. Data visualization tools would help the planetary defense community by giving them a better picture of what is going on in space and helping them respond more quickly if an asteroid or comet was headed towards Earth.

There are several tools that can be used to visualize the presence of potentially hazardous objects. One way is through data visualization, which helps us understand why things are happening, as well as compare different patterns and trends that could inform future outcomes. Another tool for visualization is 3D modeling software, which allows scientists to create simulations of potential impacts on Earth should a NEO event occur. These tools will help scientists predict where these objects, based on their size and trajectory, might land so they can prepare accordingly to minimize potential damage.

One of the most well-known uses of 3D modeling involves (Willis et al. 2015) a physical science technician at NASA's Johnson Space Center in Houston, Texas, who created a 3D model of the Stardust capsule after it returned from comet 81P/Wild 2 (Stephan et al. 2008). This 3D model gave scientists greater insight into how the material inside the capsule is arranged. It also provides more insight into what comets are made of and is helping us understand our origins as humans. 3D modeling can also be used to create visualizations of the solar system to help scientists understand how it is organized. This research also provides a 3D visualization tool to view the orbital path of selected celestial objects in a WebGL-built solar system. Chapter 4 provides in-depth detail regarding the various approaches and techniques relating to the Planetary Defense Knowledge Gateway framework.

This dissertation proposes an interoperable planetary defense knowledge gateway framework that uses data mining, information integration, and visualization techniques to solve the challenges identified earlier. The proposed framework is designed to allow planetary defense researchers to access, query, and visualize data from disparate sources using a single interface. In addition, the framework utilizes a modular approach to allow for future expansion and integration of new data sources. The modular approach also allows for different stakeholders, such as government agencies and private companies, to contribute data to the planetary defense community without having to share their proprietary data. Finally, the proposed framework includes a set of algorithms that utilize both supervised and unsupervised learning techniques to automatically classify planetary defense data. These algorithms are designed to allow planetary defense researchers to quickly identify relevant data when searching for information.

### 1.1 Objectives

The primary objective of this dissertation is to design and develop a framework to facilitate the integration, visualization, and analysis of dispersed PD information. The objective is associated with the following research tasks:

1. Conduct a comprehensive literature review to identify 1) the types of data and information that are used in PD domain, 2) existing data integration and fusion approaches, systems, and tools, and 3) current data visualization systems.
2. Design an automated data pipeline with multiprocessing capabilities for the PD domain. This will be used to capture the semantics of PD data and information, which will facilitate the integration of heterogeneous PD data sets. This dissertation attempts to use data integration techniques to merge

data from various sources, and **data fusion techniques** to transform data that are in incompatible formats.

3. Develop a prototype system that incorporates the integrated data model and utilizes existing data management approaches, systems, and tools to support the integration, visualization, and analysis of PD data.
4. Evaluate the prototype system using real-world PD data sets. This evaluation will help to assess the effectiveness of the proposed system in supporting PD decision making tasks.

## 1.2 Dissertation Organization

The remainder of this dissertation is organized as follows: Chapter 2 provides an overview of literature related to this dissertation. This chapter is divided into four sections. The first section identifies the type of data and information that are used in PD domain. The second section discusses existing approaches, systems, and tools for data integration and fusion. The third section presents current data visualization systems. Finally, the fourth section discusses existing approaches for three-dimensional trajectory data management and mining. Chapter 3 describes the design of the proposed data fusion, data model, and data framework. Chapter 4 elaborates on the details of a parallelism-related study that was conducted and describes the approach taken for celestial object coordinates accuracy verification. Chapter 5 provides a detailed description of the prototype system that has been developed to support the integration, visualization, and analysis of PD data sets. Furthermore, Chapter 5 also elaborates on the approach taken to optimize memory and GPU for smooth visualization. Chapter 6 provides a comprehensive introduction of the prototype system. Chapter 7 presents the conclusions of this dissertation and future work.

# 2. LITERATURE REVIEW

## 2.1 Types of Data and Information in Planetary Defense Domain

The United States' *National Near-Earth Object Preparedness Strategy and Action Plan* identifies possible outcomes from potential impacts by NEA, which depends on the objects' characteristics (Daou and Johnson 2019). One project related to data identification is the *Arcetri Near-Earth Object Precovery Program* (ANEOPP), which focuses on identifying NEO's in images from previous "archival materials"(Boattini et al. 2001). Another project is the NEOWISE mission (NASA), which shows data for all year 2015, outspanning the development of the first-ever set of data by combining all publicly available exposures from both the AllWISE and NEOWISE-Reactivation (NEOWISER) mission phases (Mainzer et al. 2014). In addition, ground-based planetary radar systems, such as the Arecibo Observatory and the Goldstone Solar System Radar (GSSR) facility, located in Puerto Rico and California, U.S. respectively, contribute to the high precision physical and orbital characterization of planetary bodies. From astrometry measurements to surface structure or subsurface composition, these facilities provide data for NASA projects and missions, including contributions to OSIRIS Rex, LRO, Cassini, Clementine, Mars Exploration Rovers (MERs), InSight, MESSENGER, and more (Usikov 2013).

The planetary defense community relies on data and information from multiple sources to identify, track, and characterize asteroids that could potentially impact Earth. These data include optical, radar, and infra-red observations (Binzel et al., 2002; Mainzer et al., 2011; Chodas et al., 2013). The planetary defense community also uses data from numerical simulations to study the physical effects of an asteroid impact (Chyba et al., 1993; Harris & D'Alessio, 1998; Melosh et al., 2000). Finally, the planetary defense community relies on data from impact simulations to study the regional and global effects of an asteroid impact (Atkinson et al., 1997; Lewis et al., 2001; Melosh et al., 2002).

Instruments are also a critical part of missions fundamental to perform a physical characterization of a potential impactor to mitigate the risk of near-Earth object impacts. These instruments and missions need to be included as part of a PD data source for future access to a comprehensive catalog of instruments used in past missions. These current capabilities will assist international space agencies in taking steps towards the creation of instrument advancements in the pursuit of reaching higher expectations for mitigation capabilities. One PD mission that will launch in the year 2022 is the Psyche mission (Snyder et al. 2020). The plan for this mission is to reach a unique, metal asteroid that is currently orbiting the Sun, and is located between planets Mars and Jupiter, the fourth and fifth planets from the Sun.

To guarantee the mitigation study team is well informed, scientists need a place to archive, discover, access, and integrate all the critical elements from the various funded resources. As mentioned earlier, this data is being collected by the variety of detection methods that the NASA NEO Survey employs. There needs to be a capability where all the data may be categorized and organized. Currently, there are multiple databases spread across multiple agencies in the United States, with one acting as the major center for data storage. The data is available to the global PD user community as well as the contributing facilities to promote the monitoring of existing NEOs and the discovery of new NEOs. Data storage is an important step within the mitigation process and having the necessary information accessible and easy to find is held as a high priority for this project. Across the databases, at least the linkage should be added to deploy a system of systems (SoS) framework. Table 1 shows a summary of databases hosted across the nation.

**Table 1. List of Planetary Defense Databases**

| **Database** | **Description** |
|---|---|
| Minor Planet Center | One of the worldwide locations for receipt and distribution of positional measurements of minor planets, comets, and outer irregular natural satellites of the major planets. |
| Asteroid Lightcurve DB | Listing of asteroid lightcurve parameters and other information, such as estimated diameters, phase slope parameters (G), albedos, and flawed (Wagner et al. 2009). |
| NASA Jet Propulsion Laboratory Small-Body Database | Collection of all International Astronomical Union-identified NEOs and comets. |
| Sentry Online Risk Table | Lists potential future Earth impacts events that the JPL Sentry System has detected. |

Overall, the observational capability of the NEO program is vast and diversified. These facilities also serve websites with detailed information about the specific instrumentations and methodologies. Forming strong connections with other partner organizations, such as the European Space Agency and Japan Aerospace Exploration

Agency, would result in a gain of observational capability and foster an attitude of cooperation when faced with a challenge from a NEO.

In summary, an integrated data resource discovery engine is essential for applications like the Planetary Defense Knowledge Gateway, as it encourages researchers to collaborate and share their findings with the entire PD community. It provides benefits, such as a) a secure, more reliable environment; b) an effective way of sharing information and resources; and c) providing the capability of accessing documents at any time. There are numerous data resource tools that help users to share and receive files from local computers via the Internet or a local network. These solutions can be applied to share various kinds of files, such as documents, videos, and images. Most file-sharing services have evolved into immersive collaboration platforms. Some of the biggest service providers of these services include the following: Google Drive, Microsoft OneDrive, Box, Dropbox, and SugarSync. In the education field, many institutes have used at least one form of a digital repository to provide data to the public. To illustrate, Figshare is one of the tools used by numerous organizations and institutes, such as the University of Adelaide, to preserve and share the community's research outputs, including figures, datasets, code, posters, and presentations (Singh 2011).

## 2.2 Existing Data Integration and Fusion Approaches, Systems, and Tools

With the recent global catastrophe caused by the Chelyabinsk impact in Russia, there has been a tremendous spike of interest devoted to developing data fusion strategies for our future generation PHA/PHO mitigation scientists, which will benefit tomorrow’s scientists with readily available knowledge from one primary source. As De Juan and Tauler (2019) note, data fusion allows data sets that present an enormous diversity to concatenate in terms of size and behavior. There are many reasons why it is beneficial to have data infused into one platform. De Angelis et al. (2015) discusses the importance of consistency and relevancy of data during urgent events. The researchers observe that fields geared towards epidemics are progressively based on as many diverse sources of information as possible. Although realistically, it is not cumbersome to produce outputs consistently with all relevant available data during a crisis, it can become challenging to integrate information from many heterogeneous sources of data—especially when the complexity of the model is high. Intriguing research questions in the planetary domain can be answered quicker, with higher accuracy, by incorporating data fusion techniques (Castanedo 2013). An integrated data-infused resource encourages researchers to collaborate and share their findings with the entire PD community. It provides benefits, such as a) a secure, more reliable environment; b) an effective way of sharing information and resources; and c) providing the capability of accessing documents at any time.

The planetary defense community has developed numerous approaches, systems, and tools for data integration and fusion. These approaches, systems, and tools include the Integrated Planetary Protection Knowledgebase (IPKB) (Stokes et al., 2009), the Small Body Environmental Data System (SBEDS) (Farnocchia et al., 2013), the Multi-Mission Archive at Space Telescope Science Institute (MAST) (Jenness et al., 2015), and the Center for Near Earth Object Studies (CNEOS) Information System (Tricarico, 2005).

The IPKB is a knowledge management system that was developed to support data integration and fusion in the planetary defense domain. The IPKB uses the Semantic Web Technologies (SWT) to represent data and information in the form of ontologies. The ontologies are used to reason about the data and information. The IPKB also uses the rule-based approach to infer new knowledge from the existing data and information.

The SBEDS is a data management system that was developed to support the storage, retrieval, and analysis of planetary defense. The SBEDS uses a relational database to store planetary defense data. The SBEDS also provides an interface that allows users to query the planetary defense data.

The MAST is a data archive that was developed to support the storage, retrieval, and analysis of astronomical data. The MAST stores planetary defense data in the form of images, spectra, and time series data. The MAST also provides an interface that allows users to query the planetary defense data.

The CNEOS Information System is a data management system that was developed to support the storage, retrieval, and analysis of planetary defense data (Chodas 2015). The CNEOS Information System uses a relational database to store planetary defense data. The CNEOS Information System also provides an interface that allows users to query the planetary defense data.

Bringing together heterogeneous datasets poses several conceptual and technical questions, particularly when considering different existing analytical approaches and developing appropriate metrics for evaluation. Understanding these potential challenges, along with the complexities surrounding different kinds of heterogeneity in the data, is key to creating viable data fusion approaches. Here are some of the limitations that PD-researchers need to consider when considering data fusion:

1. Semantic heterogeneity— databases shown in Table 1 are different data sets referring to the same phenomena. However, are they sharing the same "values"? NASA, ESA, and MPC use different observation satellites to capture Near-Earth Asteroid coordinates. So, "Which dataset trumps the other ones?:" This is one of the most asked questions while performing data fusion.
2. Temporal heterogeneity –data sources may be static or dynamic. Stationary data sets are snapshots of phenomena at a point in time. Dynamic data may be "streaming" data that reflects a phenomenon continuously or with larger time intervals. The interval parameters might not be the same, so finding out which interval to pick is vital for a successful fusion.
3. Modeling Heterogeneity—most of the data gathered are from sensors and devices that capture analog and digital phenomena based on an underlying model. The nature of this abstraction itself matters when fusing data to understand what one should expect when they combine the data. For instance, are the underlying assumptions compatible?
4. Infrastructure Heterogeneity—large data sets may not be captured due to storage/power/bandwidth limitations, and data may be corrupted due to infrastructure issues. Data may be incomplete due to operational and systems

issues. Some data sources are offered in XML format, while others are in JSON format. Some are provided as APIs, while for others, we might have to connect to their telnet system to retrieve their data. These are also additional challenges that we need to consider.

## 2.3 Data Visualization

According to the 2010 US National Research Council (NRC) report *Defending Planet Earth: Near-Earth Object Surveys and Hazard Mitigation Strategies* (Board and Council 2010), four broad mitigation options were identified, including civil defense, kinetic, tractor, and nuclear. Civil defense (evacuating specific areas) may be sufficient for a small object approach, but a blast deflection of some kind will be required for large objects. In between, a kinetic impactor that imparts momentum onto the NEO could be employed, or a gravity tractor that pulls the NEO over a long period could be used if there is enough warning time.

**Table 2. Short-warning mitigation techniques comparison**

| Methods | Technology development requirement | Residual issues |
|---|---|---|
| Kinetic Impact | Research the feasibility and effect of an impact | Question of whether the asteroid might be inadvertently fragmented; More warning time is required; Kinetic impactors cannot deal with as large of asteroids as NED deflection can |

| | | |
|---|---|---|
| Gravity tractor | The mass of the asteroid | Gravity tractors might not be useful for the large asteroids of over 500 meters in diameter (Bonilla 2015) |
| NED Deflection | Research the feasibility and interaction between the NED's detonation products and the asteroid that requires research | Question of whether the asteroid might be inadvertently fragmented |

In Table 2, we summarize three methods, including Kinetic Impactor, Gravity tractor, Nuclear Engineering and Design (NED) Deflection. For kinetic impact, the principle is that the NEO is deflected following a hit from an impactor spacecraft. For the gravity tractor (GT), the idea is to use a spacecraft hovering above an asteroid and relying on the small gravitational attraction to change the object's orbital path ever-so-slightly. For nuclear methods, NED deflection is conducted by nuclear explosions triggered at a distance, on the surface, provoking the ejection of rocks from the object, which, in turn, reacts by a small deflection. The first line of defense lies on remote observations for NASA's Near-Earth Object (NEO) Survey for detecting the existence of potentially hazardous asteroids. According to the Near-Earth Object Program (Yeomans et al. 2001), the NEO Survey has recognized and classified over 90 percent of NEAs that are over one kilometer in diameter. This achievement is mostly due to the varied detection methods that are engaged by the Survey and by the computational ability to process and store enormous datasets to correctly catalog the threat that each asteroid may represent to the Earth. Apart from recognizing the existence of the asteroid, these observations can also regulate and govern the essential properties of the asteroid. With the physical and chemical properties in the database, the PD community is better equipped to design an efficient mitigation plan. The purpose of observational capabilities study is to exemplify some facilities that are steering observational study and how the amplified partnership of these facilities can best inform the mitigation segment of PD. These connections and collaborations are a vital component of the architecture framework. Within the scope of this research, several methods of observation/characterization were identified: Telescopic Detection, Space Detection, and In-situ Missions.

Telescopic detection is the main method of observing and detecting NEOs. Being the most cost-effective choice, telescopic detection can provide us with tracking ability and physical characteristics at a fraction of the cost of radar systems and space telescopes. Space Observation is a method for characterizing NEOs because the telescope is in space and has a much farther reach. Very few space telescopes exist due to cost. A notable space characterization project is NASA's NEOWISE project, which allows precise approximations of their diameters, which, in turn, allows the NEOWISE team to debias diameter estimates across the detected NEO population. Although majorly for physical characterization, Space observation has detected NEOs. In-situ Missions are crucial to conducting (near) real-time research of an asteroid. The capability to land a spacecraft on a moving asteroid to collect real-time data as well as retrieve geological samples is imperative to truly understand the physical and chemical composition of asteroids.

In the US, there are currently three ground-based NEO-search programs to monitor, track, and discover NEOs via some form of visualization. In addition to that, external organizations have also contributed to the visualization field of PD to track and monitor the solar system, PHA/PHO, and the health of other planets. Some of the most highly acknowledged tools are mentioned in Table 3.

**Table 3. List of applications that provides solar system visualization services**

| Tools | Description |
|---|---|
| NASA Eyes on the SOLAR SYSTEM | Free application for the MS-Windows and MAC that lets users travel throughout the solar system and fly alongside the spacecraft—both current and historical ("NASA's Eyes," n.d.). |
| Celestia | The free space simulator for the MS-Windows, MAC, and Linux allows the user to visualize and explore the universe in three-dimension ("Celestia: Home," n.d.). |
| WorldWide Telescope | WWT offers the viewer imagery from the world's best ground and space-based telescopes, information, and stories from multiple sources, and mixes it all into an immersive media experience ("WorldWide Telescope Web Client," n.d.). |
| Universe Sandbox | Physics-based space simulator for the MS-Windows, MAC, and Linux. It merges gravity, climate, collision, and material interactions to reveal the beauty of our universe and the fragility of our planet ("Universe Sandbox," n.d.). |
| Kerbal Space Program | Space Program Simulator that allows the user to build and fly rockets and space planes, get them into orbit, and perform scientific experiments from space. During its development, NASA collaborated with KSP's developers to create an in-game mission mirroring NASA's Asteroid Redirect Mission ("Kerbal Space Program - Create and Manage Your Own Space Program," n.d.). |

Furthermore, there are numerous other data visualization systems that have been developed to support the visual analysis of planetary defense data. These systems include the Small Body Information System (SBIS) (Farnocchia et al., 2015), the Jet Propulsion Laboratory Asteroid Tracker (JPLAT) (Mommert et al., 2014), and the University of Arizona Lunar and Planetary Laboratory Asteroid Tracker (LPLAT) (Howell et al., 2006).

The SBIS is a data visualization system that was developed to support the visual analysis of planetary defense data. The SBIS uses the Google Maps API to visualize planetary defense data. The SBIS also allows users to query the planetary defense data.

The JPLAT is a data visualization system that was developed to support the visual analysis of planetary defense data. The JPLAT uses the Google Earth API to visualize planetary defense data. The JPLAT also allows users to query the planetary defense data.

The LPLAT is a data visualization system that was developed to support the visual analysis of planetary defense data. The LPLAT uses the Google Maps API to visualize planetary defense data. The LPLAT also allows users to query the planetary defense data.

### 2.4 Three-dimensional Trajectory Data Management and Mining

The issue of searching and managing two-dimensional data has been analyzed extensively. There are several publications on how to evaluate queries when the attribute values at each time t are known for sure. Emrich et al. (2012) presents a method for representing and querying uncertain spatio-temporal data in an efficient manner. The researchers further propose novel approximation methods to predict the uncertain movement of things as a follow-up to their work (Emrich et al. 2012). These strategies allow for efficient and effective filtering while query evaluation uses a hierarchical index structure. In another study (Niedermayer et al. 2013), the researchers suggest a sampling technique that employs Bayesian inference to ensure that observed paths reflect the data in the database.

These solutions are tailored to: 1) two-dimensional data that is 2) located on the surface of the earth. These existing solutions for large-scale trajectory database management and mining cannot be directly applied to stellar bodies and their trajectories. There is a lack of research or expertise on mining potentially hazardous trajectories of stellar bodies (Tsumoto and Hirano 2010).

There are two main challenges when it comes to managing and mining large sets of stellar body trajectories:

1) The three-dimensional nature of space: Most trajectory data management solutions are designed for two-dimensional data. This means that they cannot be directly applied to data that exists in three-dimensional space.

2) The movement of stellar bodies is not constrained by the surface of the earth: The trajectories of stellar bodies are not limited to the surface of the earth. This means that traditional methods for managing and mining trajectory data (such as those mentioned above) cannot be used.

Given the size and complexity of such data sets, it is not trivial to develop solutions for managing and querying them effectively (Jiang 2015). This dissertation discusses the challenges associated with managing and querying three-dimensional trajectories and propose several possible solutions.

The first challenge is because three-dimensional trajectory data are typically too large to be stored in a traditional relational database management system (RDBMS). Thus, it is necessary to use a more scalable database solution, such as a NoSQL database (Kim et al. 2020). However, NoSQL databases are often less efficient when it comes to query processing, due to their lack of support for advanced query optimization techniques (NICA et al. 2019).

The second challenge is that three-dimensional trajectory data are typically too complex to be queried using traditional SQL queries. This is because SQL was designed for two-dimensional data and does not support the type of query operations that are typically required for three-dimensional trajectory data. For example, SQL does not support queries that involve spatio-temporal computations, such as finding all trajectories that pass within a certain distance of a given point in space at a given time.

To address these challenges, numerous possible solutions have been proposed. One solution is to use a graph database, such as Neo4j (Drakopoulos, Gourgaris, and Kanavos 2018), to store and query three-dimensional trajectory data. Graph databases are well suited for storing and querying complex data structures, such as three-dimensional trajectories. They also support efficient query processing, due to their support for indexing and query optimization.

Another solution is to use a MapReduce framework, such as Hadoop, to process three-dimensional trajectory data. MapReduce is a scalable batch processing framework that is well suited for processing large amounts of data. It is also efficient, due to its parallel processing capabilities (Thusoo et al. 2009).

A third solution is to use a stream processing framework, such as Apache Storm, to process three-dimensional trajectory data (Shieh et al. 2017). Stream processing is a type of real-time data processing that is well suited for handling large amounts of data. It is also efficient, due to its ability to process data in parallel.

Each of these solutions has its own advantages and disadvantages. In general, graph databases are more efficient for query processing, but they may not be able to handle the large amount of data that is typically associated with three-dimensional trajectory data. MapReduce frameworks are scalable and can handle large amounts of data, but they are less efficient for query processing. Stream processing frameworks are efficient for query processing, but they may not be able to handle the large amount of data that is typically associated with three-dimensional trajectory data.

In conclusion, there is a lack of research or expertise available on mining three-dimensional trajectories. Given the size and complexity of such data sets, it is not trivial to develop solutions for managing and querying them effectively. However, several possible solutions have been proposed, each with its own advantages and disadvantages. Further research is needed to determine the best way to manage and query three-dimensional trajectories effectively.

# 3. DATA FUSION AND DATA FRAMEWORK

## 3.1 Data Description

The data adopted by this dissertation are listed in Table 4 with detailed information.

**Table 4. Data sources used in the study.**

| **Dataset** | **Organization & Data Source** | **Output** | **Parameters** | **Related Product** | **Role** |
|---|---|---|---|---|---|
| Small-Body Database | NASA JPL Horizons Web-Interface | Formatted HTML/Plain text/download | Settings by observer location and time span | Generate ephemerides for solar-system bodies | Data-fusion/NEO model visualizations Parameters accepted: ephemeris type/target body/location/time span/table settings/display output |
| Close Approach Data | NASA CNEOS Close Approaches | CSV/ Excel/ REST API | Nominal dist and H limit | Close approaches to the Earth by NEOs | Data-fusion/Visualization/ Knowledge base Parameters accepted: near future/distance/ H limit |
| Sentry Impact Risk | NASA CNEOS Impact Risk Data | CSV/ Excel/ REST API | Observation time/ probability/ Palermo scale/ H value | Highly automated collision monitoring system that continually scans for future impact | Visualization/ Communication Parameters accepted: Object designation/ probability/diameter/scale |
| Orbits and Discoveries | Minor Planet Center | JSON/HTML | Frequencies/ Date | Minor Planet Center orbits and discoveries online browser | Visualization |

| | | | | | |
|---|---|---|---|---|---|
| SBDB | Small-Body Database | API endpoints | Similar to NASA JPL Horizons | Method of requesting machine-readable data | Visualization/web app |
| DAMIT | DAMIT asteroid models | Tar.gz | Asteroid Attributes | 3D shapes | Visualization |
| NHAT | NASA/JPL NHATS Data API | JSON/HTML | Asteroid Attributes | Observation and Radar details | Visualization |

### *3.1.1 JPL Horizons Small-Body Database*

JPL's small-body database contains orbital elements and physical parameters for all known asteroids and most recent comets. The small-body database search engine is used to generate the required custom tables of orbital and physical data for all asteroids and comets in this dissertation. As a proof of concept, a table of orbital elements for Bennu, 67P, and PDC 2019 was generated using this tool. Output can be displayed in the browser, accessed via web API, or optionally downloaded in CSV format. To assure that this data source works with the data-fusion strategies, we created a python script to download the dataset via their provided web API connection automatically. The Small-Body Database Browser will be used to view data for a specified asteroid or comet. This tool will allow us to visualize the orbit diagram of a potentially hazardous object, its discovery circumstances, and its selected known physical parameters. Figure 1a displays the orbital elements variable, while Figure 1b shows the orbital determination parameters associated with "Bennu" retrieved from NASA JPL Small-Body Database Browser.

| Orbital Elements at Epoch 2455562.5 (2011-Jan-01.0) TDB<br>Reference: JPL 97 (heliocentric ecliptic J2000) | | | |
|---|---|---|---|
| Element | Value | Uncertainty (1-sigma) | Units |
| e | .2037451084785423 | 2.2082e-08 | |
| a | 1.126391025934071 | 4.0261e-11 | au |
| q | .8968943641658774 | 2.486e-08 | au |
| i | 6.034939533607825 | 2.8213e-06 | deg |
| node | 2.060867329373625 | 3.9444e-06 | deg |
| peri | 66.22306846088361 | 6.0706e-06 | deg |
| M | 101.7039479473255 | 2.6478e-06 | deg |
| $t_p$ | 2455439.141945784743<br>(2010-Aug-30.64194578) | 3.2088e-06 | TDB |
| period | 436.6487281348487<br>1.20 | 2.3411e-08<br>6.41e-11 | d<br>yr |
| n | .8244613502889271 | 4.4203e-11 | deg/d |
| Q | 1.355887687702265 | 4.8464e-11 | au |

(a)

| Orbit Determination Parameters | |
|---|---|
| # obs. used (total) | 518 |
| # delay obs. used | 22 |
| # Doppler obs. used | 7 |
| data-arc span | 6821 days (18.67 yr) |
| first obs. used | 1999-09-11 |
| last obs. used | 2018-05-15 |
| planetary ephem. | DE431 |
| SB-pert. ephem. | SB431-N16 |
| condition code | 0 |
| norm. resid. RMS | .27502 |
| source | ORB |
| producer | Otto Matic |
| solution date | 2018-May-19 06:45:18 |

(b)

**Figure 1: (a) Bennu's orbital elements referenced on 2011-Jan-01. (b) Orbit Determination Parameters referenced at the same Epoch**

### *3.1.2 CNEOs Close Approach Data*

Although a "close" passage astronomically can be far away in human terms (millions or even tens of millions of kilometers), near-Earth objects can occasionally approach close to the Earth. CNEOS Earth Approach software detects predicted Earth close approaches for all known NEOs, in both the past and the future, and tabulates the close approach data organized by time. The close-approaches data will be accessed via their HTTP API services. This API provides access to current close-approach data for all asteroids and comets in JPL's SBDB. The only limitation to using this data source is that by default, the query parameter is set up to only retrieve NEO less than 0.05 au (astronomical unit) in the next 60 days, sorted by date. This data source was also used in the data-fusion proof-of-concept application as an initial prototype. The sample data output is returned in JSON format, as shown in Figure 2.

```
{
  "signature":{"version":"1.1","source":"NASA/JPL SBDB Close Approach Data API"},
  "count":"2",
  "fields":["des","orbit_id","jd","cd","dist","dist_min","dist_max","v_rel","v_inf","t_sigma_f","h","fullname"],
  "data":[
    ["2007 JB21","9","2418800.878283280","1910-May-09 09:05","0.0020925812637796","0.000330379338764904","0.00489001
    ["2012 BX34","16","2419429.176816497","1912-Jan-27 16:15","0.00224445877558233","0.00107169260360805","0.0791138
  ]
}
```


**Figure 2. Sample JSON output retrieved from CNEOS close-approach API endpoints**

### *3.1.3 CNEOs Sentry Impact Data*

Sentry is a highly automated collision monitoring system that continually scans the most current asteroid catalog for possibilities of future impact with Earth over the next one hundred years. Whenever a potential impact is detected, it will be analyzed, and the results will be published to the Sentry database. A summary of all known potential impacts can be found on the main Sentry page ("Sentry: Earth Impact Monitoring," n.d.). The table quantifies the risk posed by the tabulated objects, using both the Torino Scale, which was designed primarily for public communication of impact risk; and the Palermo Scale, which was designed for technical comparisons of impact risk. A Palermo Scale value less than zero and a Torino Scale value of zero indicate a risk below the background level, which is the average risk from the entire NEO population. This data source will be essential to the visualization tool, as the objects with the highest impact probability are retrieved from this database first before automatically scanning the other data sources for additional visualization information. This source provides the following information: object designation, year range, potential impact, impact probability, velocity, magnitude, estimated diameter, Palermo scale minimum, Palermo scale maximum, and Torino scale. Table 3. Describes the field outputs that will be used in this dissertation from the Sentry Impact Risk DB.

**Table 5. Sentry Risk Impact field outputs a summary**

| Field | Units | Description |
|---|---|---|
| des | | Primary designation of the object. |
| method | | Analysis method used: `LOV` = Line-of-Variations, `MC` = Monte Carlo. |
| fullname | | Full name/designation of the object. |
| ps_cum | | The cumulative hazard rating according to the Palermo technical impact hazard scale, based on the tabulated impact date, impact probability and impact energy. |
| ps_max | | Maximum hazard rating according to the Palermo technical impact hazard scale |
| ts_max | | Maximum detected hazard rating according to the Torino impact hazard scale, based on the tabulated impact probability and impact energy. The Torino Scale is defined only for potential impacts less than 100 years in the future. |
| ip | | The cumulative probability that the tabulated impact will occur. The probability computation is complex and depends on a number of assumptions that are difficult to verify. For these reasons the stated probability can easily be inaccurate by a factor of a few, and occasionally by a factor of ten or more. |
| n_imp | | Total number of potential impacts resulting from this analysis |
| energy | Mt | The kinetic energy at impact: 0.5 × `mass` × `v_imp`$^2$, measured in Megatons of TNT. |
| h | | Absolute Magnitude, a measure of the intrinsic brightness of the object. |
| diameter | km | This is an estimate based on the absolute magnitude, usually assuming a uniform spherical body with visual albedo of 0.154 (in accordance with the Palermo Scale) but sometimes using actual measured values if these are available. Since the albedo is rarely measured, the diameter estimate should be considered only approximate, but in most cases will be accurate to within a factor of two. |
| mass | kg | This estimate assumes a uniform spherical body with the computed diameter and a mass density of 2.6 g/cm$^3$. The mass estimate is somewhat more rough than the diameter estimate, but generally will be accurate to within a factor of three. |
| v_inf | km/s | Relative velocity at atmospheric entry neglecting the acceleration caused by the Earth's gravity field, often called the hyperbolic excess velocity. |
| v_imp | km/s | Velocity at atmospheric entry. |
| pdate | | Date and time (UTC) used in the Palermo scale computation. |
| cdate | | Date and time (Pacific Local) of the impact analysis computation. |
| first_obs | | Date and time (UTC) of the first observation used in the analysis. |
| last_obs | | Date and time (UTC) of the last observation used in the analysis. |
| darc | d | Number of days spanned by the observations used the the analysis. |
| nobs | | Total number of observations used in the analysis. |
| ndel | | Number of radar delay (range) observations used in the analysis. |
| ndop | | Number of radar Doppler observations used in the analysis. |
| nsat | | Number of optical satellite observations used in the analysis. |

### *3.1.4 Minor Planet Orbit and Discovery Data*

The MPC discovery circumstances datasets will be used to visualize the discovery details for minor planets numbered (1) to (5000)—a list of all provisional designations belonging to a specific minor planet that can be found using Minor Planet Center's interactive designation converter ("Minor Planet Designations," n.d.). This dataset returns the discovery date of an object in YYYY MM DD form and Name Ref. (the reference to the citation accompanied by the naming of the object). An example of a returned citation for 5000 is shown in Figure 3.

**Citation for (5000)**

The following citation is from *MPC* 19342:

```
(5000) IAU = 1987 QN7
     Named in honor of the International Astronomical Union, established in
1919 as the international association of professional astronomers.  Most of
its activity is concentrated in some 40 commissions, each devoted to some
particular area of astronomical or astrophysical research.  At a competition
held in July 1991 among members of Commission 20, "Positions and Motions of
Minor Planets, Comets and Satellites", for the selection of a name for the
five-thousandth minor planet to be given a permanent number the name "IAU"
received the largest number of votes, including that of the discoverer.
```

**Figure 3. Minor Planet Center Interactive Designation Converter returning citation for IAU (5000).**

### *3.1.5 SBDB*

The SBDB Database is a small-body database that provides a way of requesting machine-readable data on a specified small body within JPL’s SSD/CNEOS Small-Body Database. The SBDB includes object identification and naming information, orbital data, and selected physical data for all known asteroids and comets within the solar system. It also contains rich ancillary data such as close approach and virtual impactor information. All the orbits are computed by JPL’s Solar System Dynamics (SSD) group. The SBDB provides complete data for each object, that is, from the designation and orbit number to physical properties. Data may be accessed in real-time or by requesting a bulk download of an entire region or all regions. In addition to positional and photometric measurements, the database includes normalized radar cross-sections (RCS) and effective optical diameters for most objects. Also provided are some ancillary data (e.g., diameter-to-diameter ratios, albedo measurements), along with information on close approaches and virtual impactors. These objects can be defined either by a specific SBDB designation or by a JPL-designated identifier. For example, the object designated "433 Eros" is also known as 1969 NA and 206039, while 99942 Apophis is also known as 2004 MN4 and 2006 SQ372. The positional data are part of a space-based frame and include (but are not limited to) Earth-centered inertial coordinates in the J2000 frame, with a nominal epoch of J2000.0; heliocentric ecliptic coordinates; and B1950 ecliptic coordinates. Orbital data include orbital elements that describe both the orbit shape and the argument of perihelion. For some objects, shape models are available as well. Photometric data include standard photometric parameters such as total magnitudes and colors in various filters and bands and the number of every certain type of observation (e.g., radial velocity measurements, occultations, eclipses). Physical properties may also be included: diameters, masses, albedos, densities, and radar cross-sections. The physical properties are commonly derived from the photometric data; however, they may also be provided in tabular form or as modeled values. The orbit data products include the orbital elements (e.g., semi-major axis, eccentricity) of an object's orbit around the Sun. It is most often implemented as a database, although it may also be delivered in other forms (e.g., an FTP site). The data are available periodically or by request for bulk download. Most of such orbits have been computed at the Minor Planet Center (MPC) using very precise techniques, making use of all observations on hand. However, some orbits are too incomplete for this method to be used. All other small bodies that meet the size and orbit characteristics for cataloged status but do not have sufficiently well-defined orbits are considered unconfirmed until their orbits can be computed. Orbit determination using various techniques continues throughout the lifetime of each object, and gaps in data coverage may result in temporary unconfirmed status (see: IAU Resolution 6).

#### ***3.1.6 DAMIT***

DAMIT is a database that consists of asteroid models that were derived using the light-curve inversion method developed by (Kaasalainen et al. 2004; Kaasalainen and Torppa 2001), combined, in some cases, with other inversion techniques. Light-Curve Inversion is a technique in the field of astronomical photometry. It involves taking a light curve from an observed asteroid and using data from other sources to generate a possible model for an asteroid's shape. The inversion process starts with a light curve observed for a particular interval, typically spanning one or two orbital cycles. The observed magnitude at each point in this light curve is then inverted into the brightness that would be observed if another hypothetical asteroid was rotating in place on that arc. This gives rise to an "inverted" light curve which can then be compared to the original light curve. Inversion is sometimes called "intensity mapping," although intensity mapping typically refers to the use of a single light curve, with intensity mapped at each point rather than brightness.

DAMIT uses the following technologies: web apps, MATLAB scripts, IDL scripts, web services, all currently running on Linux servers. DAMIT allows the user to select any object in the database and output detailed information about it. DAMIT contains detailed information for 3303 asteroids with 5715 models and four tumblers with known light curves, rotation periods, and amplitudes. The asteroid data are downloadable in FITS format, while the light-curve parameters are available via web services. A brief description of each model is also recorded in DAMIT's database, including information about which papers published its creation. DAMIT provides a unique interface to search for objects by the period range and other search criteria.

DAMIT is a project of the IOTA Asteroid Lightcurve Data Exchange (ALDEx). ALDEx provides access to light-curve data that have been obtained for asteroids and comets using ground-based telescopes. Access to light-curve data is provided via web services. The DAMIT interface allows the user to select any object in the database and output detailed information about it. As an example, below are some images of Phaethon taken with GMOS at Gemini North, including an image of the asteroid's light curve obtained with the Gemini Multi-Object Spectrograph (GMOS) and a graphical representation of its rotation period.

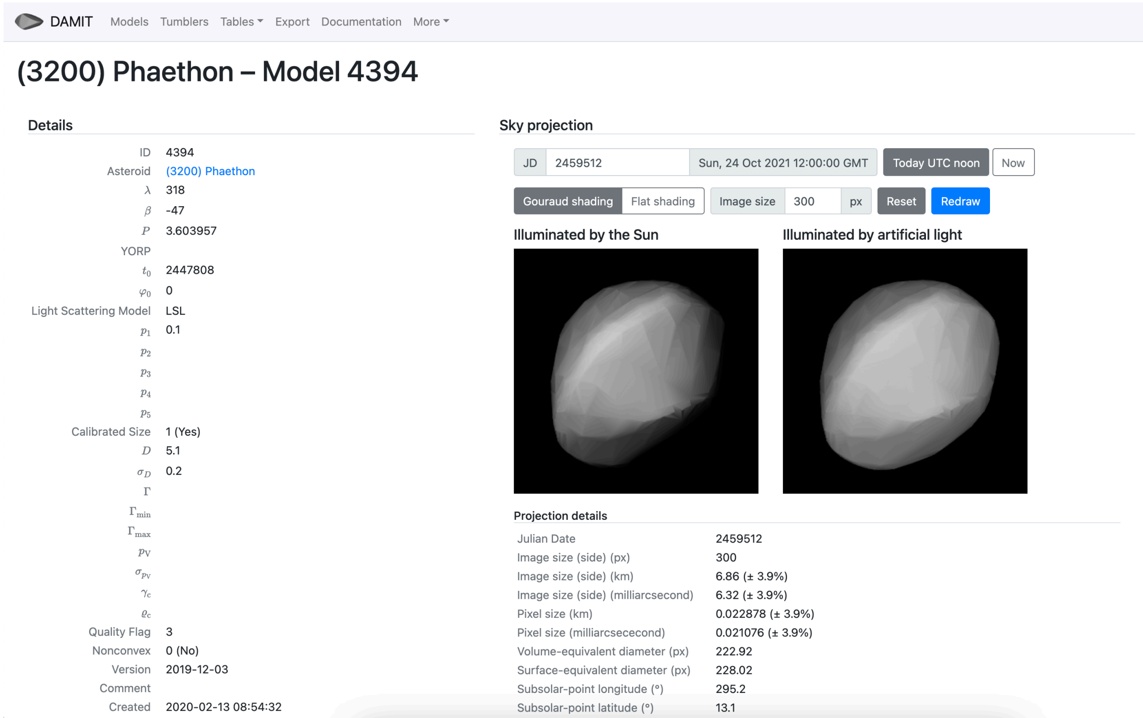



**Figure 4: Visualization of 3200 Phaethon 3D render in the DAMIT Platform**

### *3.1.7 NHAT*

NHATs stands for Near-Earth Object Human Space Flight Accessible Targets Study. This study began in September 2010 by the Planetary Science Division at NASA Headquarters in Washington, D.C., U.S. The purpose of the project is to identify any known NEOs, particularly NEAs, that might be accessible by future human space flight missions. To monitor these asteroids, (Barbee 2014) developed a system that locates and analyzes new NEAs and those with updated orbits from the JPL SBDB. This automated process executes daily, and the results are distributed to the astronomy community and the general public to aid in the prioritization of telescope time allocation. The initial phase of the study was independently performed by the Goddard Space Flight Center and Jet Propulsion Laboratory (JPL) (Perozzi et al. 2010). These objects are accessible with a single Hohman transfer, meaning it would take less than six months for an object to travel from low Earth orbit (LEO) to the object to be visited.

The first phase of the project included a complete survey of all NEAs whose orbits could be accessed by future human space flight missions, as well as NEAs that had been detected up through 2013. The objects were then divided into different access classes based on how much delta-v it takes to rendezvous with them. The classes range from "easily accessible" (Shang and Liu 2017) to reachable only with a "major technical feat" (Farquhar et al. 2002; Perozzi, Rossi, and Valsecchi 2001). Almost all of the objects in NHATS are in access-class 1 or 2, meaning they can be reached by a vehicle that is already going to a NEO and uses less than about 10% of its fuel. NHATS identified 17 NEAs that fall into access-class 1 and 2. Phase 1 was validated by both JPL and GSFC independently with very similar answers. Phase 1 found there to be about two dozen NEA accessible by human space flight missions, depending on the technology used, such as solar electric propulsion (SEP) and ion rockets. The study also found that access-class 1 and 2 objects were more numerous than any other, with about half of the NEAs in NHATS coming from classes 1 and 2.

A study conducted by Barbee, Mink, and Adamo (2011), showed that here are three things that a NEA must do before it can be considered a NHATS qualifying NEA: 1) Earth departure C(sub 3) energy < = 60 km(exp 2)/s(exp 2); 2) total mission delta-v < = 12 km/s (including an Earth flyby for a total delta-v of 12 km/s); 3) and a maximum estimated size > = 30 m. Of the 765 NEAs which passed the Phase II trajectory filter, a total of 590 NEAs also satisfied the further constraint of maximum estimated size > = 30 m. The NHATS study found that Atiras (aphelion < 0.983 AU), Atens (aphelion > 0.983 AU, alpha < 1.0 AU), Apollos (perihelion < 1.017 AU, alpha > 1.0 AU), and Amors (1.017< perihelion < 1.3 AU) orbit families are the most accessible with only 11% of known Apollos passing the NHATS qualification criterion, 31% of known Aten's passing, 456% of known Apollo's passing, and 116% of known Amor's passing. These simple statistics demonstrate that NEAs like Atens (aphelion > 0.983 AU, alpha < 1.0 AU) orbit possesses a feature that makes them accessible to human space flight missions like the Comet Surface Sample Return (CSSR) and Near-Earth Asteroid (NEA) Scout missions.

In a recent study conducted by (Michel et al. (2016)), the researchers were able to produce the first photometric result of the survey started in 2013. The purpose of the study was to find out NEAs' rotational periods, light curve amplitudes, and shapes. Two other centers also performed the first phase of the NHATS survey for validation purposes.

### 3.2 Method and Dataset Relationship

It is critical to have a domain ontology, or at least a vocabulary repository, to provide a common understanding of specific domains that can be communicated between people and applications. This is especially true for the planetary defense community because it is highly inter-disciplinary when it comes to knowledge integration and mitigation. Due to the lack of established ontology for planetary defense knowledge integration, we constructed a vocabulary repository of 146 concepts describing the semantics of the information related to NEO observation, NEO characterization, NEO impact modeling, and decision support and mitigation. These vocabularies range from sample NEOs (e.g., Bennu) to observatories (e.g., Arecibo Observatory), from impact modeling (e.g., airburst modeling) to disruption strategies (e.g., NED Deflection). Vocabularies are initially sorted by number and alphabet when displayed with abstracted description. Most descriptions are statements with references, as discussed in the methodology section for vocabulary repository construction, which is listed at the end of the glossary so that users can trace back to the original links or publications for detailed information. The search function is enabled to find specific vocabularies, as indicated in Figure 5a. The result shows all relevant records and not only entries that contain keywords in the phrase, but also those that involve the keywords in their descriptions.

Ontology associated with Smart Search supports related search suggestions, offering convenience for users to explore other possible keywords from the vocabulary repository. As Figure 5b demonstrates, related searches for "model" are "VARIATIONAL_ANALYSIS," "PHYSICS BASED_MODEL," and "HIGH_FIDELITY_SIMULATION," etc. All relatives are followed by a score indicating relevancy which, based on Yang et al. (2017), is pre-calculated for the relations among vocabularies within PD ontology. The general process of generating these correlations is to establish a pairing relationship between every two words using PD OWL and use scores to indicate the closeness between the two. As Figure 9b illustrates, the weight for "model" and "high_fidelity_simulation" is 0.75, which was used for ranking the related search results in Figure 5a.

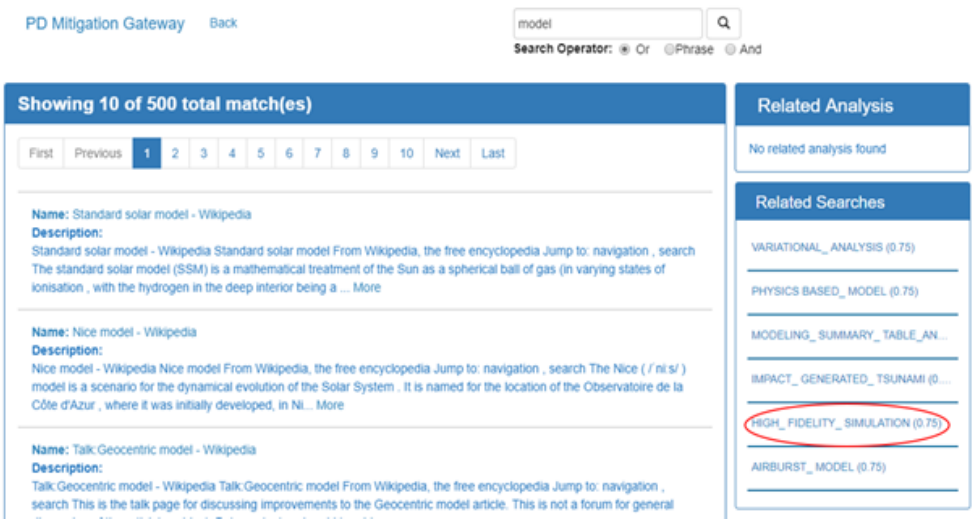



**Figure 5a. Search results of "Bennu" within integrated glossary for knowledgebase ontology.**

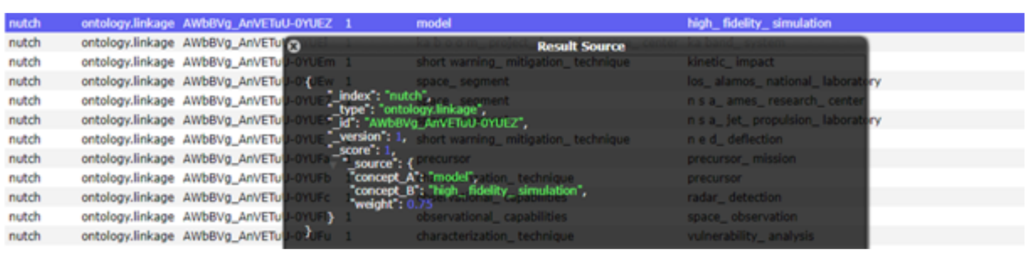



**Figure 5b. Example showing Related Searches calculated for "model."**

The proposed framework in this dissertation offers an interactive three-dimensional visualization tool. As a proof-of-concept, this tool used precise data fusion methods written in python to merge asteroid data from JPL Horizons, ESA S2P, and

NASA 3D Resources. Firstly, the object was fetched from the JPL Horizons Small-Bodies telnet system, via our python script, using the usage command shown in Figure 6. After downloading the object-related contents, another python script gets triggered to parse the downloaded NASA JPL data into JSON format. Then the data gets stored in the local typical data service instance. Additional steps will need to be taken to integrate the other data sources into the data fusion method. Figure 8 shows a quick architectural overview of the future data fusion process.

```
Usage: python scrape.py [id] [starting_tbd] [ending_tbd] [output]
       id:            body ID (ex: 399)
       starting_tdb:  YYYY-MMM-DD (ex: 1995-Dec-25)
       ending_tdb:    YYYY-MMM-DD (ex: 2000-Mar-25)
       output:        output interval (ex: 1d, 1h)
       ex: python scrapte.py 399 1995-Dec-25 2000-Mar-25 1d
       ** this would return data on Earth (399) between Dec. 25 1995 and Mar. 25 2000,
       and the coordinate data would be shown 1 day **
```

**Figure 6. Connection to JPL Horizons Small-Bodies DB**

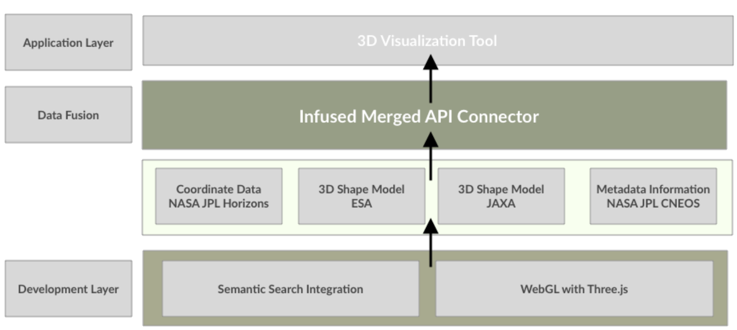



**Figure 7. Architectural overview of the data-fusion process**

The six primary data sources stated in Table 2 had to be incorporated into the unified data storage platform. These data sources had several commonalities that were taken under consideration during the data collection step. Commonalities, such as numbered id, object name, and citation code, were common across all organizations since all observations need to be submitted to Minor Planet Center first. The MPC is responsible for the designation of minor bodies in the solar system: minor planets, comets, and natural satellites. The MPC is also responsible for the efficient collection, computation, checking, and dissemination of astrometric observations and orbits for minor planets and comets. Once the numbered id was retrieved, several actions were needed to access related data from different sources, some of which only allowed the system to collect the data via FTP/SFTP; other sources were accessed via API or direct web download. Manual data collection would have been vastly time-consuming since there are over 26,000 registered near-Earth objects, with new objects getting discovered daily. An automated route had to be taken that would capture the object id and follow several preformulated instructions to collect the data in an organized manner. To get to that stage, a database had to be architected with all the necessary tables that would host the data. Figure 8 visualizes the UML diagram that was associated with the unified database.

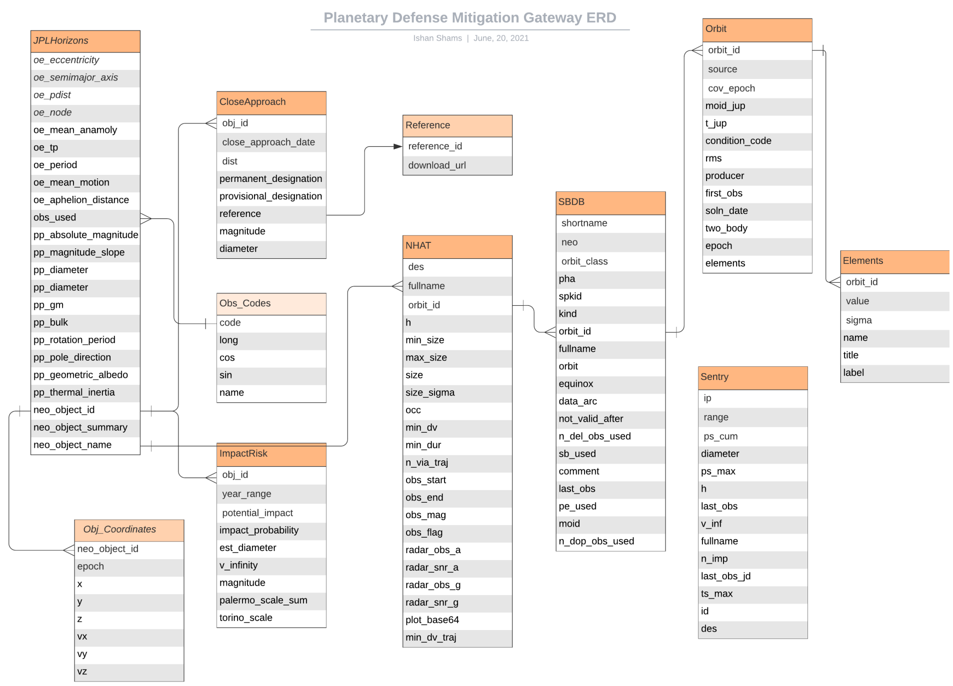


**Figure 8: Planetary Defense Knowledge Gateway ERD**

## 3.3 Considered Data Fusion Parameters

Data fusion is the process of combining data from multiple data sources to provide a complete understanding or representation of the data (Qi et al. 2020). This dissertation discusses the multi-data source data fusion methodology, which uses combined analysis on different types of data to determine most physical and surface properties (De Juan and Tauler 2019). An advantage of using multi-data methodology is that one can characterize an asteroid in a single coherent inversion that yields remarkable results. The use of this technique eliminates the need for multiple independent analyses to derive accurate parameters, such as shape and rotation state, about an asteroid's orbit and orientation (i.e., spin axis). Multi-data sources also allow the characterization of asteroids with different shapes and rotation states. Furthermore, this method uses more data sources to derive a complete set of physical properties.

The study of properties of near-Earth asteroids (NEAs) provides great insight to the understanding of the early Solar System formation and evolution as well as the threat potentialities to life on this planet (Warner and Stephens 2019). In general, using multi-data sources, such as interferometry radar, to determine the size, mass, and spin state of a NEA allows us to understand its composition. Understanding the composition of NEAs can lead us to determine their origin and provide insight into the early Solar System formation and evolution (Reddy et al. 2015). The study of these properties also allows us to predict potential impact events that may occur in the future. Studying an asteroid's composition that has been affected by space weathering gives us a better understanding of the effects of space weathering on its potential impact on energy. A multi-data source data fusion methodology that is presented in this dissertation is the combined analysis of wind, and light curve disc resolved images (Nachouki and Quafafou 2008). The use of this methodology can yield highly accurate estimates on shape, spin state, and albedo by eliminating the need for multiple independent analyses. The major parameters that can be derived using this methodology are the asteroid's size, mass, and rotational state.

Radar data is used to determine an oblate spheroid shape of asteroids by combining the results from both direct Doppler delay measurements and interferometry data (Cox 1972). The use of these different techniques allows for an oblate spheroid shape to be determined. It has been found that radar interferometry yields a higher quality data set in determining an asteroid's properties than direct Doppler delay measurements alone.

Using the Light-Curve Inversion method from the DAMIT dataset, hundreds of asteroids with different shapes and sizes have been derived using multi-data sources, such as interferometry and wind images, to determine the most accurate results (Ďurech, Sidorin, and Mikko Kaasalainen 2010). The use of multi-data sources allows for a more efficient data analysis that can yield highly accurate physical properties about an asteroid's shape, rotation state, and albedo. Combined analysis on different types of data to derive physical properties provides a better understanding of these asteroids' origin, composition, and evolution.

Shape modeling consists of two processes: Finding the rotation state (polar orientation) and orientation (spin axis) of an asteroid (Hanuš et al. 2011). An NEA's rotation state is calculated by analyzing its geometry represented with light curve disc resolved images (Ďurech et al. 2016). Once this is done, it can be rotated to align to an astrometric model with an azimuth angle measured from the west (az = 90°). If this rotation state doesn't match any of the rotation states modeled from the light curve disc resolved images, it is considered as a previously unknown rotation state.

Orientation can be derived using at least three different methods: stellar occultation timings, mid-infrared thermal radio matching, and optical interferometry techniques. Timing of an NEA's stellar occultations yields an accurate estimation of an NEA's size and shape. The most accurate results are possible when using a statistically significant number of stellar occultations. For example, in one study, results indicated that for an NEA with an albedo of 0.15, the standard deviation on its half-light radius is approximately 2%. The mid-infrared thermal radio matching method provides a consistent estimation of shape and size. An important factor in the mid-infrared thermal radio matching method is the constraining factor on an NEA's rotation state (Kaasalainen, Ďurech, and Sidorin 2014). However, this method can only be used for asteroids that have either a spheroidal shape or triaxial ellipsoidal shape, since it requires an assumption of symmetry. In another example, an NEA with an albedo of 0.15 and a triaxial ellipsoidal shape with dimensions of 20×8×8 km was determined to have an obliquity between 30° degree and 40°, the spin axis between 330° degree and 120°, the longitude of ascending node (λ) between 40° degree and 270°, an argument of perihelion (ω) between 0° degree and 10°, and right ascension of the ascending node (β) between 60° degree and 110° degrees. The optical interferometry technique is sensitive to the orientation of an NEA's axis-of-symmetry and spin axis.

The determination of an NEA's albedo is important since it allows us to determine the effect of space weathering on its potential impact energy and predict its orbital evolution. An NEA's surface properties such as color, composition, and texture can be derived from the study of multi-band photometry. The photometric colors of an NEA are usually measured in the visible wavelength range (0.35–1.05 μm), while its thermal properties are derived by observing the mid-infrared wavelengths (3–100 μm). It is important to note that analyzing multi-band photometry data on its own may not be able accurate since optical brightness does not directly correspond to the object's thermal properties. However, if both photometry and thermal data are available, it is possible to determine the albedo of an NEA.

Albedos for Main-belt Asteroids (MBA) and Near-Earth Asteroids (NEA) derived from spaceborne observations, such as Wide-field Infrared Survey Explorer (WISE), NEOWISE, Spitzer and Herschel Space Observatory, have been published in the NASA/JPL Planetary Data System.

The size, shape and rotational state of a NEO can also be derived from ground-based observations, such as radar or Optical Interferometry. In addition to photometric data, NEAs reflect radio waves at frequencies between 1-20 MHz. The presence of a NEA's radio emission can be attributed to the different mechanisms that produce it. One mechanism is electron cyclotron maser emission caused by electrons accelerated in an electromagnetic field during a single or multiple interactions with plasma waves. Another source of this radiation is free-free emission, which arises from a collision between a fast magnetosonic wave and a slower ion-acoustic one. The last mechanism for this emission is non-thermal bremsstrahlung in which an energetic charged particle interacts with the ambient plasma and gains energy in during its collision. Note that observations in frequencies above the decimeter range provide more information about an NEA's surface properties than those at lower frequencies since the electromagnetic wave is attenuated by the surface as it propagates.

Disk-integrated photometry is the most abundant source of data on NEOs. This observation method is most often used for NEAs with an albedo of 0.07 or higher, which are detectable at optical wavelengths even during the night. For period determination, a single lightcurve covering the full rotation is sufficient. However, a set of such lightcurves at different phase angles is required to determine the asteroid's shape, spin state, and spectrum. Kaasalainen and Torppa's (2001) Light-Curve Inversion method has been reviewed in numerous studies. Since then, the method has been widely used, and several improvements were introduced by various researchers (Ďurech et al. 2016; Kaasalainen, Ďurech, and Sidorin 2014). These studies are available at the Database of Asteroid Models from Inversion Techniques (DAMIT) (Ďurech, Sidorin, and Mikko Kaasalainen 2010). The reliability of method was proved by comparing its results with independent data, such as laboratory asteroid models (Müller et al. 2005), adaptive-optics (AO) images (Pravec et al. 2006), stellar occultations (Ďurech, Sidorin, and M. Kaasalainen 2010), and (433) Eros (Kaasalainen, Torppa, and Piironen 2002).

### 3.4 Data Pipeline

Data pipelines are a set of data-processing steps that are built on top of each other so that the output of one operation feeds into the next downstream operation (Quemy 2019). It transforms and organizes raw observation data from multiple sources into forms that can be ingested by analysis tools. The goal of this process is to efficiently integrate data sources used in this research. The data pipeline is an important aspect for many researchers in the planetary sciences. It collects, assembles, and manages datasets for use by tools that detect orbital features. To efficiently integrate data sources from CNEOs JPL, Sentry, NHATs, DAMIT, Minor Planet Center, the system uses an analysis pipeline written in bash. In summary, this processes individually attributed asteroid data that are later inserted into the PostGRES SQL database. The Data pipeline uses reduction techniques, and other methods for performing quality trimming, read mapping, and Baseline data filtering that are discussed in this section.

The data pipeline has 4 different stages. The downloading stage has its bash script, which performs certain tasks to the input dataset at each stage. The first step is sbdb_data.sh, followed by close-approaches_data.sh, sentry_data.sh, nhats_.sh, and finally DAMIT.sh. Each stage calls the script that is beneath it in an orderly fashion, which is shown in Figure 9.

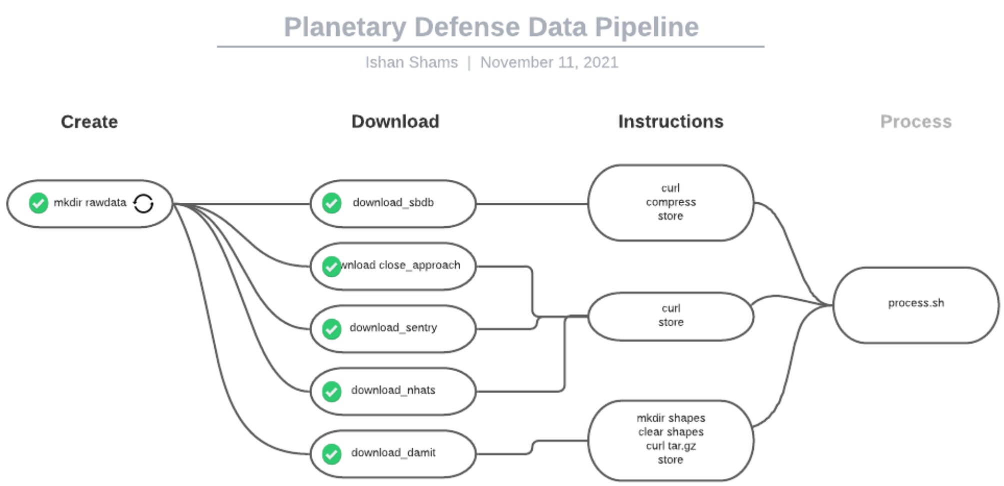



**Figure 9. Data Pipeline Diagram**

There are a few of the data reduction techniques that are used in bash data pipelines. The first is quality trimming. In this method, some of the values of the measured quantities have been discarded from both the input and output datasets to minimize ambiguity and increase the precision of the database. This reduces noise for downstream processing. The read mapping technique is a process of mapping a set of reads to their respective locations on a sequence alignment for future analysis on the sequence alignment (Xin et al. 2013). For this pipeline, read mapping involves mappings from our input file to our aligned file, scanned on two-reads files as well as on single-reads files which are then saved as positional mapped files. Baseline data filtering is another technique that has been implemented in Bash data pipelines. This method of filtering is applied to the baseline, which is a set of observations over time of an object passing through Earth's atmosphere. The idea behind this technique is that objects with low albedos will be brighter in the baseline dataset because of atmospheric scattering while high albedo objects will be darker (Harris et al. 1989). These outliers must then be removed before post-processing.

The first step of the data pipeline is to pass our input file (SBDB_Query.api) with the following fields, shown in Table 6, into the download_sbdb.sh bash script. This script calls the API, then zips the output data inside the rawdata folder that gets generated during the Create phase of the pipeline. This script performs a three-way merge of the data from all the remaining data sources into a single file and adds an additional column to each row which is a unique identifier for each asteroid. This is then piped into the next stage that downloads data from CNEOs Close Approach dataset.

**Table 6. Parameters passed to each data sources**

| Data sources | Parameters |
|---|---|
| SBDB | full_name,pdes,name,class,neo,pha,moid,moid_jup, epoch,e,a,q,i,om,w,ma,tp,per,n,ad,first_obs, last_obs,n_obs_used,H,M1,diameter,density, extent,rot_per,GM,pole,albedo,BV,UB,IR,spec_T,spec_B |
| Close-Approach | Date-min, date-max, dist-max, full_name |
| Sentry | * (all) |
| NHATs | * (all) |
| DAMIT | * (all) |

The second, third, and fourth step of the data pipeline is to pass our input from close_approach, sentry, and NHATs datasets into the download bash script. This script calls the respective APIs with cURL. The main.py Python script performs post-processing on the Sentry data files, merging the data into multiple files in rawdata.

The fifth step of the data pipeline is to pass the DAMIT input URL into the DAMIT_data.sh bash script, which calls the DAMIT/scripts/DAMIT_downloader.py Python script. The DAMIT_downloader.py Python script downloads all 3-Dimensional asteroid model files from the website for this project and saves them in a directory called shapes/.

The final step of the data pipeline is to pass our input file (DAMIT_output/assets/) into a final sh script called DAMIT_final.sh which calls the DAMIT/scripts/DAMIT_preprocessor.py Python script, whereby it performs some post-processing on all the asteroid models. The final product is saved as rawdata/shapes.

### 3.5 Data Models

This research was mainly built with consideration around loosely coupled design principles. Future researchers would be able to leverage the current system's models to integrate new data sources. It would be possible to write a new model that extracts data from a different source without having to rewrite a lot of the existing code. The Django framework also allows for easy integration with advanced machine learning algorithms. It also encourages good programming practices, such as writing readable and well-commented code. Django has carefully designed many of its components to be pluggable, meaning that they can be swapped out for alternative implementations without changing the surrounding code (Alchin, Kaplan-Moss, and Vilches 2013). One of Django's most significant decisions is to use the template system as its primary I/O layer. This decision significantly increases the freedom Django programmers have when constructing their applications; it does not tie them to any database or data output format. Django uses the database layer as its primary storage mechanism (Holovaty and Kaplan-Moss 2009). This layer is completely agnostic on how data gets displayed; it typically relies on an external system (such as the Django template system) to “render” data back into HTML.

Django models are the data structures, processes, and various tooling for managing content types in Django; they are the way Django interacts with databases. Models are also designed to be separate from views (template layer) (Alchin et al. 2013). Django treats models as regular Python classes. These classes allow encapsulation of both data and the various operations that may be performed upon it. Each model maps to a single database table. Each row in the database table is represented by an instance of the appropriate Django model class, and each field on the model corresponds to a column in that table. One can define custom methods and foreign keys for models. Django has built-in support for many common field types and provides mechanisms to define custom field types. When a model is created, Django creates a table for it with some fields automatically. One can use in_bulk() to query objects from certain models and sort through all of the objects of the target model. Default objects are loaded into memory when needed.

As default, Django loads only what one needs, but if querying is done then objects are loaded into memory completely. Django provides a few base classes that are helpful when building custom models. Django models rely on a few primitive behaviors from Python, such as defining custom constructors and deconstructing objects into fields. Every model class inherits from the Model base class, which provides default behavior common to all models. When one wants to add or edit data in tables, they can use Django forms. Forms are used for both editing existing data and for creating new data in the database.

The Planetary Defense Knowledge Gateway framework leverages seven custom models. These models are ObjectType, OrbitClass, SpaceObject, CloseApproach, SentryEvent, NHATSObject, and ShapeModel. The ObjectType model accepts Enum parameters that are either an asteroid or a comet. Comets are further classified into three types: long period, Halley type, or Encke-type. The OrbitClass model is designed to capture the orbit classification of an object in its type and subclass. For example, if the object orbits around the Sun, between the inner region of an asteroid belt and near Jupiter's orbit, then it is considered a Main-Belt Asteroid. SpaceObject is a model that captures information about near-Earth objects stored in SBDB. This model identifies and stores object name, orbit class, object type, basic orbital elements, and diameter. The CloseApproach model captures close approach of near-Earth objects with the Earth. For this model, one must input a date and time as well as details about velocity and distance between near-Earth and its nearest point of closest approach to the Earth. The SentryEvent model is designed to capture information on objects that have been flagged by Sentry as potential impact threats. This model captures information, such as object name, Sentry Object of Interest number, and Sentry Risk Assessments. NHATS Object model is designed to capture the NASA-designated near-Earth objects that will become candidates for close missions during the next launch window and has a predicted encounter date less than 20 years in the future. This model captures information, such as object name, NHATS Object of Interest number, and NHATS identification date. The ShapeModel is designed to capture information on shape models through its type and subclass. For example, irregular (oval cross-section), spherical (perfect sphere), conical (cone-shape with half-angle less than 60 degrees), and contact binary. Figure 10 shows a visual representation of the hierarchical structure of the data model designed for the Planetary Defense Knowledge Gateway framework.

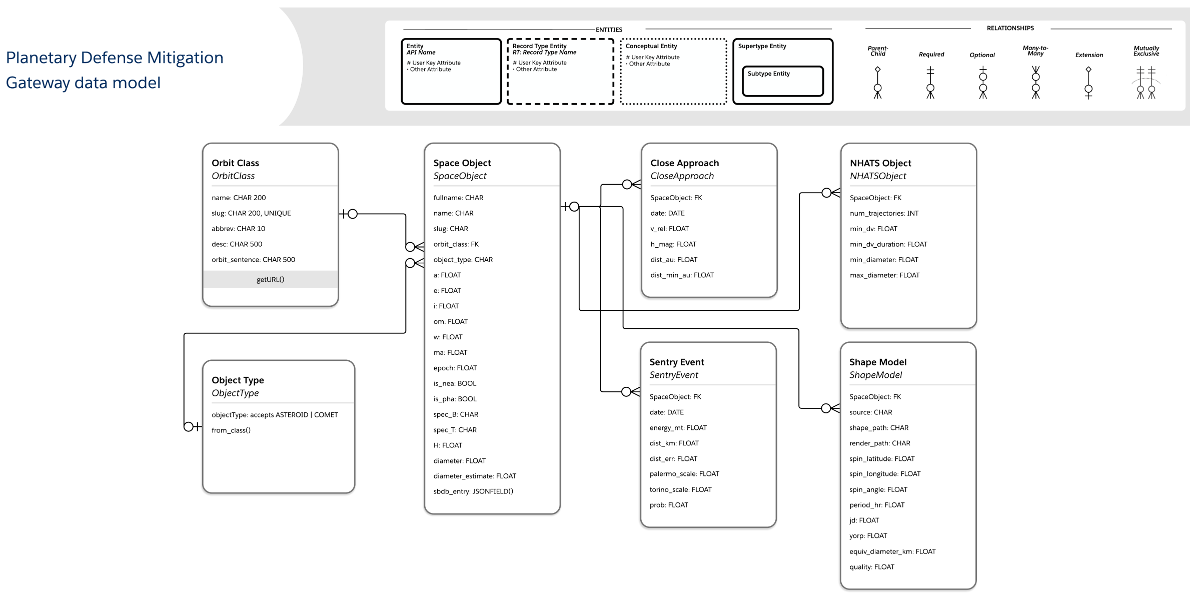
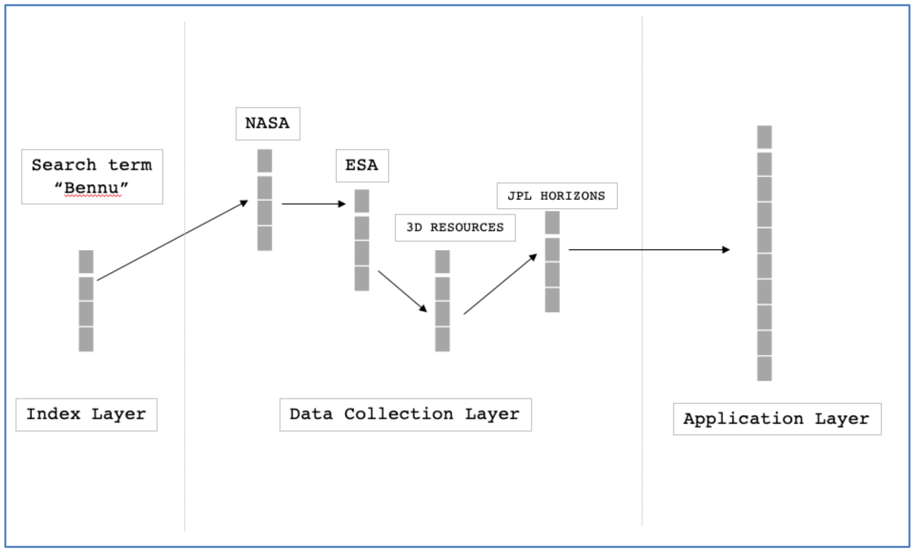



**Figure 10: Django Data Model Design**

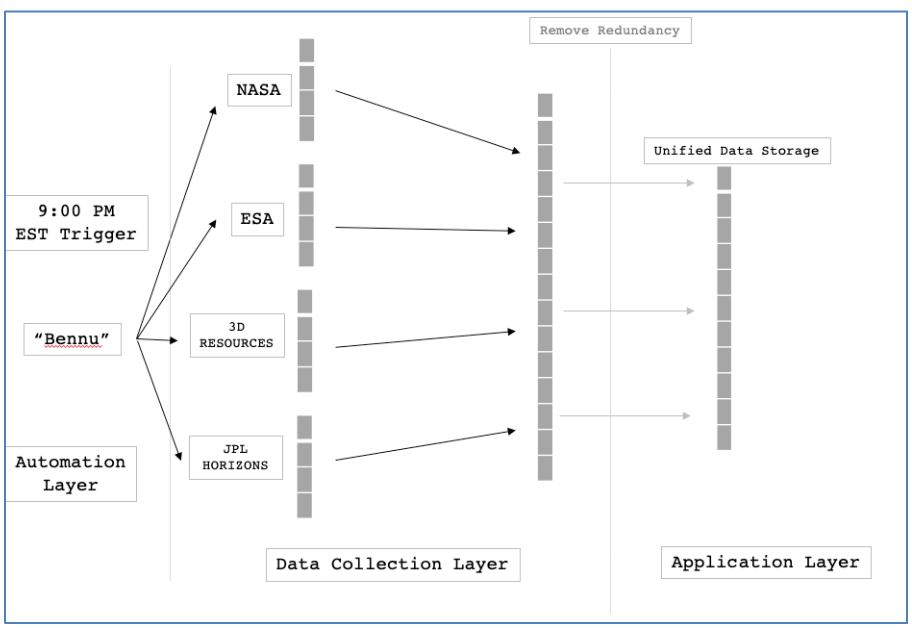
The framework leverages Django Admin capability which allows the administrator to access and manage database content through a visual interface. The Planetary Defense Knowledge Gateway framework's model classes accept all the available options as described in the ObjectType, OrbitClass, SpaceObject, CloseApproach, SentryEvent, NHATSObject, and ShapeModel objects. The framework also provides a few pre-defined descriptions that return detailed object diameters composition and classes so one can visualize the data to determine if it is an impactor or not. This framework uses multiple inheritances for its model classes, so changes are reflected across all inherited models. This way, model classes can be modified in one place, and it has a cascading effect on all models.

### 3.6 Result

For research topics that rely on data sources that are scattered amongst different organizations, it is difficult for systems to return data quickly. Traditional ways involve working with one database at a time to retrieve the dataset required to proceed to the next data source. This causes a delay since the data calls are being made in a synchronous fashion. A better approach is to use a data structure that relies on a scheduled automation tool to retrieve related datasets daily and upload them into one unified data storage system. In this dissertation, a system was designed that leveraged a bash data pipeline with an intricate Django solution incorporated with Postgres Database. Figure 11 shows a visual difference between a traditional approach and a unified data structure solution. The planetary defense community will find this method of storing and retrieving data useful because it will allow for quicker access to the data that is necessary for research. In addition, the planetary defense community can use this system to visualize the data in a way that is helpful for understanding the potential threat of an asteroid. By using this system, the planetary defense community will be able to take advantage of a) increased retrieval speed for planetary defense research, b) reduced need for manual intervention, and c) scheduled execution for data pipeline.




(a)




(b)

**Figure 11 Synchronous Storage and Unified Storage: (a) synchronous storage with multi-data source indexing, (b) unified storage with data retrieved via automated asynchronous indexing**

# 4. PARALLELING THE FUSION PROCESS USING PYTHON

In the last few decades, we have seen a major shift in terms of computing power and capabilities. Computers and other devices that we use to compute information are now incomparably more powerful than they were just a decade ago. This change has been largely driven by changes in how processors work internally as well as an increased focus on parallel processing—which is what allows computers to do more things at once. Parallel processing systems allow for tasks, such as image recognition, data mining, rendering graphics, and other similar tasks, to be broken up into smaller parts and processed simultaneously so that they can be completed much faster than if these tasks were done one after another sequentially (in series).

Traditional scientific application software, on the other hand, is typically sequential, with each process occurring one after another. This limits the amount of parallelism that a program can utilize because there is no way for a task to begin until its predecessor has completed it. Parallel programming presents an exciting new opportunity to not only make use of the full potential of hardware resources, but also to rethink how we write our scientific applications.

Given the opportunity for such a paradigm shift, scientists and engineers need to be aware of how efficiently their current programs can be parallelized. Not only because it is an important part of putting high-performance computing (HPC) to work but also because parallel computing results in less code that needs to be written and maintained, making the life of a scientist easier.

In a variety of astronomical disciplines, parallel computing has been utilized. Supercomputers have been adapted to handle computationally demanding simulation code and physical modeling. N-Body simulations of massive star and galaxy clusters, radiative transfer, plasma simulation around pulsars, galaxy formation and mergers, and cosmology are just a few examples (Singh, Browne, and Butler 2013). However, many of the astronomical image processing and complicated data analysis activities are still done serially. One of the reasons for this is the inherent and perceived complexity of parallel programming. Another explanation could be that day-to-day astronomical data processing activities do not take an exceptionally long period to complete. Despite this, there are a few parallel modules dedicated to astronomical image processing.

In this chapter, parallel data processing techniques was explored to parse and process the data that feeds the Planetary Defense Knowledge Gateway. To execute more complex astronomical activities at a faster rate, a few instructions for parallel processing on multicore machines was proposed. In addition, this chapter also discusses parallel data processing and the numerous alternative solutions available. It also focuses on the implementation process behind the Planetary Defense Knowledge Gateway data processing. Furthermore, three different astronomical data processing examples are benchmarked.

### 4.1 Parallel Data Processing

In general, parallel code is much more complex than serial code. Debugging is an issue only for parallel programs where many processes depend on results from other processes. However, parallel processing of enormous datasets is not an issue. Moving to parallel coding not only requires engineers to have a proper set of hardware and software tools, but also requires them to rethink how they write their code. A parallel program requires multiple cores or computational nodes to execute. The first thing that comes to mind when attempting to solve a problem in parallel is how to divide it up into subtasks that may be handled simultaneously.

In general, task parallelism vs. data parallelism are two methods for achieving parallelization. Each computing node may execute the same or diverse code in parallel in the case of task parallelism. The input data is distributed across the computing nodes, and the same code processes the data elements in parallel in data parallelism. Data parallelism is easier to implement and more appropriate for most astronomical data processing applications. This research chose data parallelism as the primary data processing technique. Given an N-processor or computing node system, the speedup that may be obtained by dividing a problem across many processors (compared to a single processor) is:

**Equation 1**

$$S = \frac{T_1}{T_N}$$

Correspondingly, $T_1$ and $T_N$ are the code runtime for one and N processors. $T_N$ is dependent on the number of computing nodes and the fraction of serial code. The total runtime of parallel code can be expressed using Amdahl's Law (Gustafson 2011).

**Equation 2**

$$T_N = T_S + \frac{T_P}{N} + T_{sync}$$

Where $T_s$ depicts the execution time of the serial fraction of the code, $T_p$ represents the runtime of code that can be parallelized, and $T_{sync}$ is the time for input/output operations. The efficiency of the parallel code execution depends on the level of code optimization. As such, a lower fraction of serial code would result in better efficiency. If we can keep N constant, then we can achieve better performance by either increasing the fraction of parallel code or decreasing the synchronization time.

Multiprocessing has been used in this research instead of multithreading to achieve parallelism. Threads are sections of code that the operating system can schedule. The operating system creates the appearance of running many threads in parallel, but it switches between them rapidly (time-division multiplexing). In the case of multicore machines, threads operate on separate cores at the same time. In distinction to multiple threads, multiple processes are defined as having distinct memory and state from the master process that calls them (multiple threads utilize the same state and memory). Parallel programming is most often done in C, C++, and FORTRAN. However, interpreted languages like Python, Perl, and Java have provided software adaptations. This chapter focuses on Python as the major language of choice with the main objective focused on scientists' time, ease of use, and code reusing abilities.

**4.2 Multiprocessing in Python**

Python supports both multi-threading and multiprocessing. The operating system manages the scheduling and switching of the threads, rather than Python's interpreter.

Python has a feature called the GIL (Global Interpreter Lock) that restricts the number of active threads to one, even if numerous cores or processors are available (GIL Python n.d.). As soon as the running thread releases the GIL, it can perform I/O operations or interpretive period checks. During this time, waiting threads may execute for a few milliseconds. This hurts multi-threaded applications, resulting in longer execution times. The performance of Python on multicore machines deteriorates because the Python interpreter wants to run a single thread at a time, whereas the operating system will schedule the threads across all available processor cores.

A more convenient way to achieve parallel execution is to utilize Python's built-in multiprocessing module. Parallelization could also be achieved by vectorizing the computations in NumPy(NumPy n.d.). Vectorization is a more efficient and optimized technique of replacing explicit iterative loops in the Python code. However, not all functions in Numpy or SciPy can be parallelized. To carry out this study, my co-researchers and I utilized the multiprocessing native module. For comparison, we've created parallel code to process the application's multi-source, integrated dataset. Although there are several multiprocessing methods to distribute tasks, this paper only focuses on shared memory or symmetric multiprocessing (SMP) techniques(Kota and Oehler 2005). The first approach was to use the multiprocessing Pool/Map (multiprocessing - Process-based parallelism n.d.) class, and the second was to create individual processes using the Process class. The following sections go through how these two techniques were used in detail.

The Pool/Map Technique: Out of the two native multiprocessing approaches, the Pool/Map technique is simpler to implement and execute. A Pool/Map approach generates a pool of worker processes and returns a list of items, as depicted in Figure 12a. With the built-in map function, a function can be applied to every iterable item in Python, as shown in Figure 12b. The map function is extended to the multiprocessing module and can be used with the Pool class to execute worker processes in parallel, as shown in Figure 12c. The import function adds the multiprocessing module into the procedure, mp.count_cpus gets the number of CPUs on the system, Pool creates a pool of ncpus processes, and Pool's map method iterates over the input element list in parallel, mapping each element to a worker function.

```
# Iterative function
def worker(indata):
    ...
    return result

# Input dataset divided into chunks
arr = [in1, in2, ...]

# Loop over arr items and append results
results = []
for item in arr:
    results.append(worker(item))
```

**Figure 12a: Example of a typical Python iterative function**

```python
#Iterative function
def worker(indata):
    ...
    return result

# Input dataset divided into chunks
arr = [in1, in2, ...]

# Iteratively run func in serial mode
results = map(worker, arr)
```

**Figure 12b: Example of a Python Map function**

```python
#Import multiprocessing module
import multiprocessing as mp

#get number of processors on the machine
ncpus = mp.count_cpus()

#define pool of ncpus worker processes
pool = mp.Pool(ncpus)

#start ncpus pool of worker processes in parallel
#output is appended to results python list
results = pool.map(worker, arr)
```

**Figure 12c: Pool/Map multiprocessing code**

The Process/Queue Approach: Although the Pool/Map technique is simpler to implement, it only allows one argument as an input parameter. If one wants to send multiple arguments, then there are two ways it can be done: packing arguments in a tuple or use process class in conjunction with a queue. An example Python code listing for Process/Queue approach is shown in Figure 13. The Process class is used to start parallel processes, and input data is placed on the send queue for processing in smaller amounts. Each worker process selects the next piece of data on the send queue after completing the previous one. The output result is put on the receive queue, and then read at the end for post-processing. Notice the argument 'STOP,' which is called to stop running processes. During the benchmarking phase, we have identified that the Process/Queue technique performs better than the Pool/Map approach in certain conditions.

```python
# Import multiprocessing module
import multiprocessing as mp

# Worker function
# iter is standard python built-in function
def worker( s_q, r_q, arg1, arg2, ... ):
    for value in iter( s_q.get, 'STOP' ):
        ...
        r_q.put( result )

# Get number of cores on the machine
ncpus = mp.count_cpus()

# Create send and receive queues
send_q = mp.Queue()
recv_q = mp.Queue()

# Start ncpus number of processes
for i in range( ncpus ):
    mp.Process( target = worker, args = ( send_q, recv_q, arg1, arg2, ... ) )

# Put input data chunks on send queue
# indata is python list of input data set,
# which is already divided into smaller chunks
    for chunk in indata:
        send_q.put( chunk )

# Get output from receive queue
results = []
for i in range( len( indata ) ):
    results.append( recv_q.get() )

# Stop all the running processes
for i in range( ncpus ):
    send_q.put( 'STOP' )
```

**Figure 13: Process/Queue Implementation in Python**

### 4.3 Asteroid Accuracy Verification

Although the calculations are handled by the SpaceKit.js library, to verify the accuracy of the data being passed to the framework, it is crucial to calculate the positions of the planets, the orbital elements, and physical attributes of the objects. The following section explains how to calculate the positions of the major planets. Positions of other celestial bodies such as comets and asteroids were also computed for accuracy-check. The heuristics were based on the following three principles: (1) redundant data, (2) contradictory data, and (3) anomalous data. Redundant data is data that is present in more than one source. Contradictory data is data that contradicts other data in the same or different sources. Anomalous data is data that does not follow the expected pattern.

The heuristics were applied to the observational data, catalog data, and expert knowledge. The observational data was verified using the duplicate detection heuristic, which identifies duplicate data based on a combination of the source, object ID, and observation time. The catalog data was verified using the duplicate detection heuristic and the consistency heuristic, which checks for consistency between the data in the different sources. The expert knowledge was verified using the relevance heuristic, which checks whether the expert knowledge is relevant to the task at hand. The tool indicated timescales in Julian Day Number. The observation EPOCH time from Horizons was saved, and the difference against real-time Julian Day Number was calculated. The time, as well as the orbital elements, was utilized to compute the planet's position indicated in

Tables 7 and 8.

**Table 7: List of orbital elements**

| Type of orbital elements | Variables | Description |
|---|---|---|
| Primary orbital elements | N | Longitude of the ascending node |
| | i | The inclination to the plane of the Earth's orbit |
| | w | Argument of perihelion |
| | a | Semi-major axis, or mean distance from Sun |
| | e | Eccentricity |
| | M | Mean anomaly |
| Related orbital elements | w1 | Longitude of perihelion |
| | L | Mean longitude |
| | Q | Perihelion distance |
| | P | Orbital period |

**Table 8: Orbital elements of the Sun and the other major planets.**

| | **N** | **i** | **w** | **a** | **e** | **M** |
|---|---|---|---|---|---|---|
| Sun | 0.0 | 0.0 | $282.9404 + 4.70935 \times 10^{-5} \times d$ | 1.000000 *(AU)* | $0.016709 - 1.151 \times 10^{-9} \times d$ | $356.0470 + 0.9856002585 \times d$ |
| Mercury | $48.3313 + 3.24587 \times 10^{-5} \times d$ | $7.0047 + 5.00 \times 10^{-8} \times d$ | $29.1241 + 1.01444 \times 10^{-5} \times d$ | 0.387098 *(AU)* | $0.205635 + 5.59 \times 10^{-10} \times d$ | $168.6562 + 4.0923344368 \times d$ |
| Venus | $76.6799 + 2.46590 \times 10^{-5} \times d$ | $3.3946 + 2.75 \times 10^{-8} \times d$ | $54.8910 + 1.38374 \times 10^{-5} \times d$ | 0.723330 *(AU)* | $0.006773 - 1.302 \times 10^{-9} \times d$ | $48.0052 + 1.6021302244 \times d$ |
| Mars | $49.5574 + 2.11081 \times 10^{-5} \times d$ | $1.8497 - 1.78 \times 10^{-8} \times d$ | $286.5016 + 2.92961 \times 10^{-5} \times d$ | 1.523688 *(AU)* | $0.093405 + 2.516 \times 10^{-9} \times d$ | $18.6021 + 0.5240207766 \times d$ |
| Jupiter | $100.4542 + 2.76854 \times 10^{-5} \times d$ | $1.3030 - 1.557 \times 10^{-7} \times d$ | $273.8777 + 1.64505 \times 10^{-5} \times d$ | 5.20256 *(AU)* | $0.048498 + 4.469 \times 10^{-9} \times d$ | $19.8950 + 0.0830853001 \times d$ |
| Saturn | $113.6634 + 2.38980 \times 10^{-5} \times d$ | $2.4886 - 1.081 \times 10^{-7} \times d$ | $339.3939 + 2.97661 \times 10^{-5} \times d$ | 9.55475 *(AU)* | $0.055546 - 9.499 \times 10^{-9} \times d$ | $316.9670 + 0.0334442282 \times d$ |
| Uranus | $74.0005 + 1.3978 \times 10^{-5} \times d$ | $0.7733 + 1.9 \times 10^{-8} \times d$ | $96.6612 + 3.0565 \times 10^{-5} \times d$ | $19.18171 - 1.55 \times 10^{-8} \times d$ *(AU)* | $0.047318 + 7.45 \times 10^{-9} \times d$ | $142.5905 + 0.011725806 \times d$ |
| Neptune | $131.7806 + 3.0173 \times 10^{-5} \times d$ | $1.7700 - 2.55 \times 10^{-7} \times d$ | $272.8461 - 6.027 \times 10^{-6} \times d$ | $30.05826 + 3.313 \times 10^{-8} \times d$ *(AU)* | $0.008606 + 2.15 \times 10^{-9}$ ~~E-9~~ $\times d$ | $260.2471 + 0.005995147 \times d$ |

To compute the planet's position in 3-dimensional space, the framework used the following equations:

**Equation 3**

$$xh = r * (\cos(N) * \cos(v + w) - \sin(N) * \sin(v + w) * \cos(i))$$
$$yh = r * (\sin(N) * \cos(v + w) + \cos(N) * \sin(v + w) * \cos(i))$$
$$zh = r * (\sin(v + w) * \sin(i))$$

The following formula was used to convert *N_Epoch* to N (today's epoch), where the epoch is expressed as a year with fractions.

**Equation 4**

$$N = N_{Epoch} + 0.013967 * (2000.0 - Epoch) + 3.82394E - 5 * d$$

## 4.4 Results and Experiments

Parallel processing could resolve a variety of astronomical data issues. Out of which, one was chosen for benchmarking purposes. To evaluate the benchmark scores, three devices with various configurations were utilized. The configuration of the machines utilized in this test is shown in Table 9. One of the machines was overclocked to 2.66 GHz and was running the Ubuntu 16.10 operating system. The 2017 MacBook Pro core i7 was running Monterey OS. Lastly, two cloud VMs were used, running

Ubuntu 16.10, with a similar setup to the first machine. For the astronomical data processing, SBDB and DAMIT data were used.

**Table 9: Hardware and software configuration of the benchmark testing machines**

| Machines | Processor | Memory | Operating Systems |
|---|---|---|---|
| Desktop Machine | Intel Core i5 Quad-Core 2.66GHz | 16 GB | Ubuntu 16.10 |
| Macbook Pro | Intel Core i7 2.9 GHz Quad-Core | 16 GB | MacOS Monterey |
| Azure VM | Standard_D4 8 CPU cores | 28 GB | Ubuntu 16.10 |
| Azure VM | Standard D96s v5 96 vCPUs | 284 GiB | Ubuntu 16.10 |

Coordinate Transformation Problem: Two built-in functions were used, specifically, xyz2rd() and rd2xyz(), to transform cartesian coordinates to sky coordinates, and vice-versa. These functions can only process transformation at a single given time. Processing these transformations, for over a million records retrieved from the SBDB database serially, is not an efficient approach. Therefore, the functions were re-written for multicore machines to process a large input of datasets in parallel. These updated functions leveraged modules such as: *multiprocessing* and *Parallel Python*.

These functions read input files with either (x, y, z) coordinates or (RA, DEC) values, and the output transform coordinates. The speedup factor, as shown in Figure 14 for xyz2rd() and rd2xyz(), is plotted against the number of processes. For this test, over one million input coordinates were fed into the program with guided scheduling. We identified that the best performance was achieved when the number of processes defined in the script equaled the number of cores on the machine. The Standard_D4 8 Core Azure VM showed better speedup than both other machines combined. As the number of processes increases beyond the number of cores, the speedup factor flatters out.

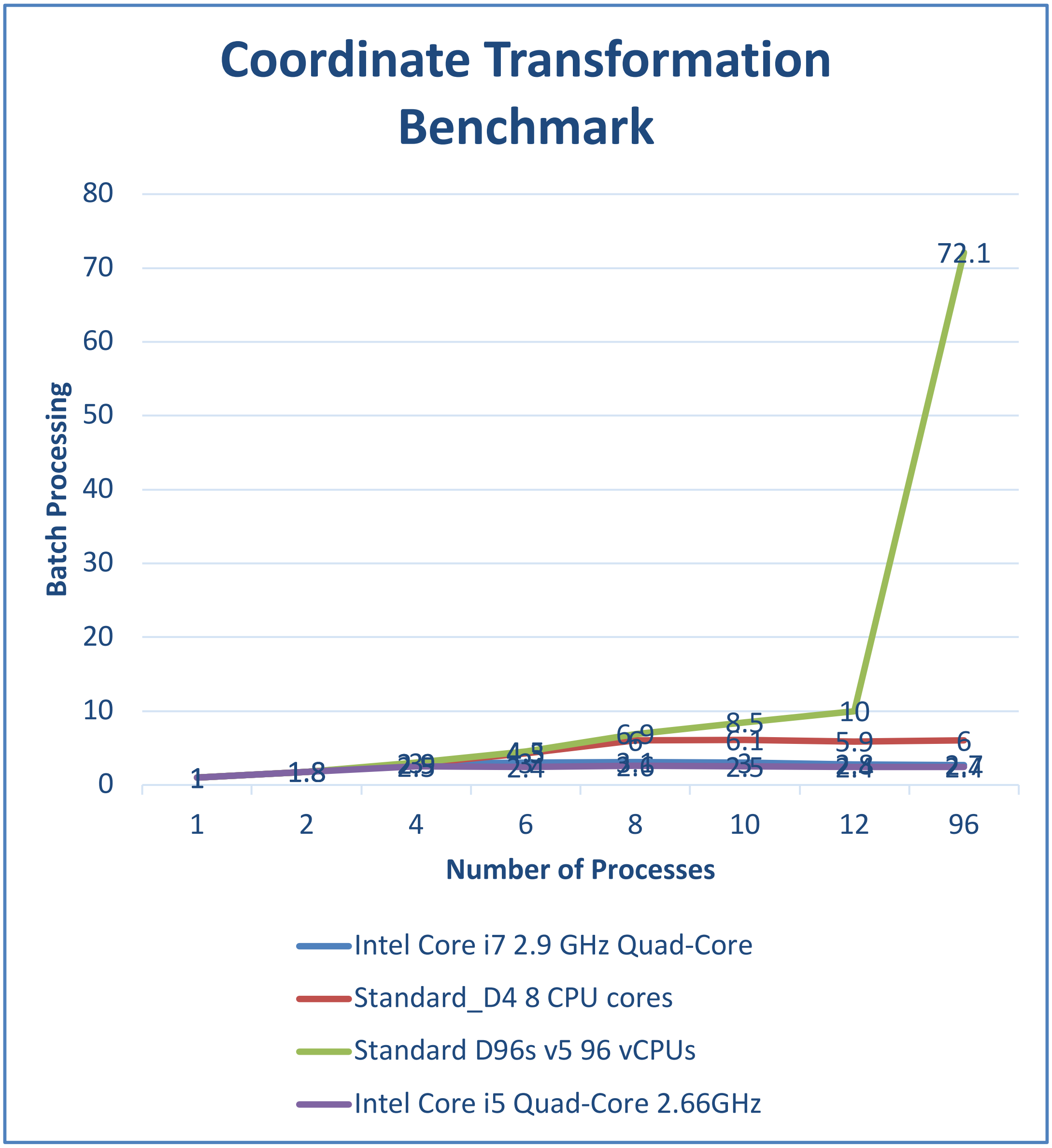



**Figure 14: Coordinate transformation benchmark.**

Downloading dataset benchmark: The stated machines were also used in conjunction to test how long it takes to download 1.2 million objects from numerous sources from start to finish. As shown in Figure 15, by applying batch processing techniques, downloading datasets can be drastically sped up with powerful machines such as the Standard D96s v5. It took approximately 31 minutes to download and process data completely. On the other side of the spectrum, it took approximately 4 hours and 28 minutes to achieve the same result when the Intel Core i5 machine was used to conduct the study.

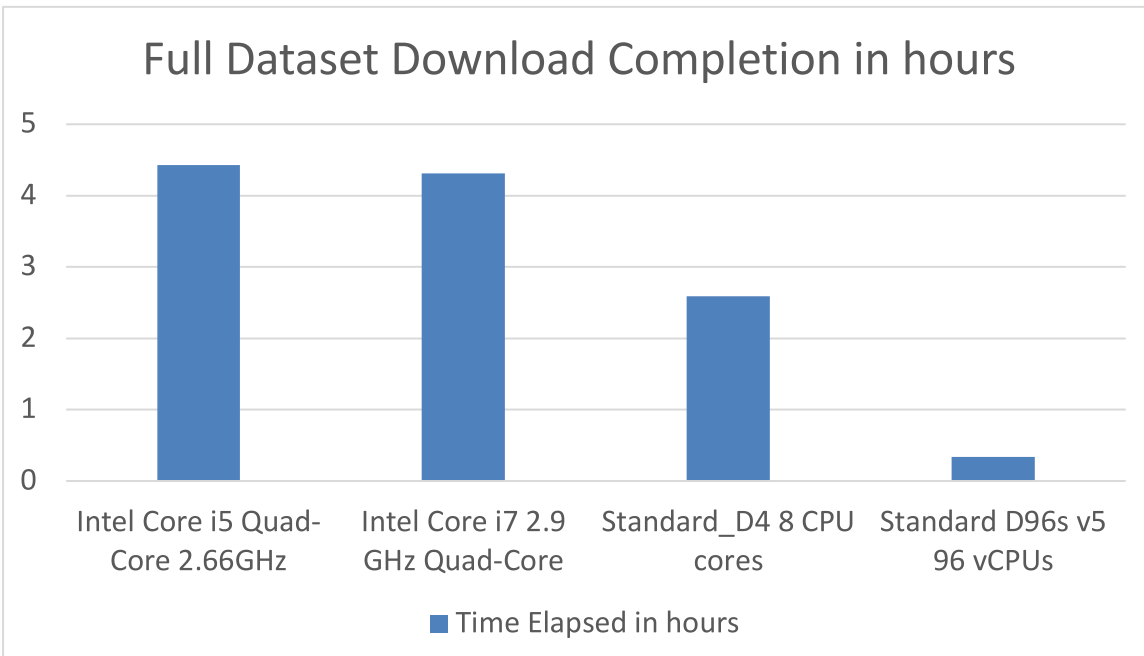



**Figure 15: Downloading full dataset benchmark.**

There are numerous different kinds of web APIs that are used today by many different organizations, but there are some universal API design principles and patterns that have been agreed upon as the most efficient ways to build APIs. There are three common styles of designing an API: RPC/SOAP, REST, and Hypermedia. While RPC (Remote Procedure Call) is more efficient for single calls (for example, if you need to get the current status of your account), it's not very good at providing a standardized communication framework for multiple resources (such as retrieving all of your account's transactions) (Bu n.d.). Hypermedia provides that capability but lacks the standardization of media formats (Bornman and von Solms 1993). The REST API design pattern rather than SOAP or RPC-based Web services include flexibility, speed, efficiency, modifiability, and ease of use (Sohan et al. 2017).

The name “REST” stands for Representational State Transfer, which describes how data can be transferred from one location to another in an internet environment. With RESTful APIs, HTTP requests are used to GET, PUT, POST, or DELETE data (henceforth referred to as HTTP verbs), and URIs (or Uniform Resource Identifiers) are used to identify data. The Planetary Defense Knowledge Gateway processes and feeds RESTful APIs as the primary method of data distribution. Since data is the main component of the APIs, in this chapter, the architecture will be introduced first, followed by API endpoints and descriptions.

Before diving into the API endpoints and descriptions, it is important to present a high-level architecture of overall data flow. It supports data manipulation to the backend PostGRES database. It also supports the application layer.

Figure 16 shows the overall architecture with emphasis on the RESTful API Web Service component. API definitions are necessary to implement communication between an API and its clients. These definitions can be written in YAML or Java, but they both have issues that prevent them from being truly universal. First, YAML allows you to describe your API, but it doesn't come with a standard that ensures backward compatibility. Second, Java is the most common language used for describing APIs, but it's simply not very human-readable. The best option is to use the Django REST framework, a standard Python library that provides the developers with a lot of tools for building APIs. Django REST Framework (DRF) is an API framework that provides a convenient, powerful, and flexible way of developing APIs. It is loosely based on Django's class-based generic views, so it's easy to understand for people familiar with Django. DRF aims to simplify the development of Web APIs by reducing boilerplate and enforcing sensible defaults. It does this through an intuitive URL routing system, which allows your API URLs to mirror the structure of your app's models. This makes DRF well-suited for building modern hypermedia RESTful APIs that are compatible with mobile clients, single-page applications, and other web clients.

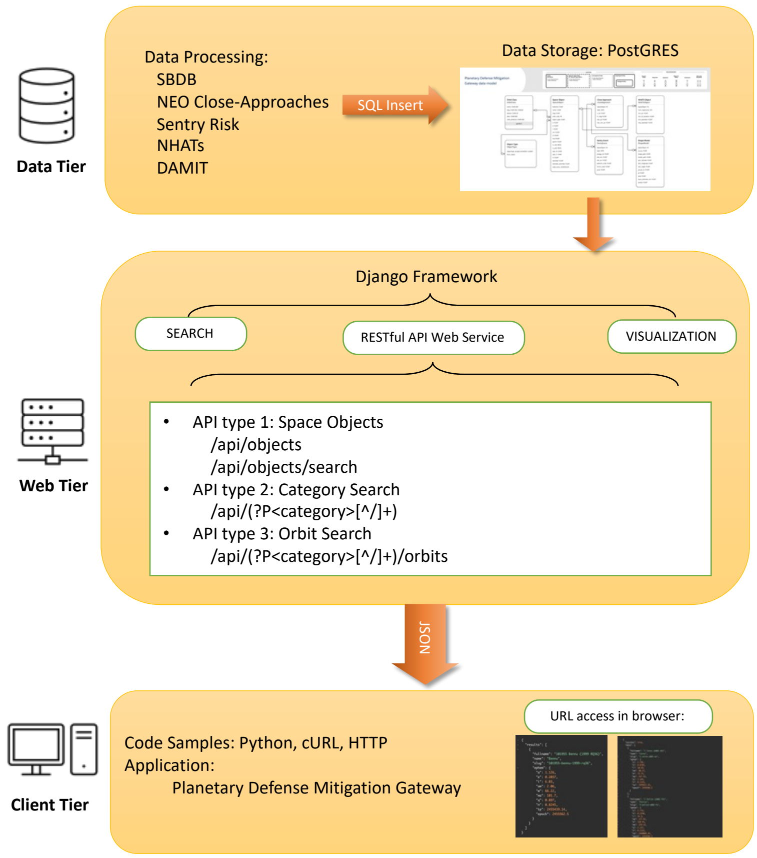



**Figure 16: Planetary Defense RESTful Web Service Architecture**

To serve the purpose of benefiting both the Planetary Defense Knowledge Gateway and the external Planetary Science community, a search web interface is also created that will be covered in Chapter 4. This interface leverages Django framework to keep the consistency and independence. In a DRF API, resources are modeled as Django models and can be specified using the class-based generic views. The API class is defined in the app's API module and defines two methods: GET, which is used to return data from the resource; and POST, for handling client input related to creating new data on the server. Defining endpoints for RESTful APIs is done by implementing a single function in each API. My co-researchers and I exposed GET to the public, and restricted POST requests only to the administrators of our application. The exposed APIs and the data passed via Django views are shown in Table 10.

**Table 10: APIs via Django Views and Associated Data Flow**

| Status | Type | URL | Description |
|---|---|---|---|
| Exposed | API | /api/objects/<slug> | Returns a list of NEOs as a JSON response |
| Exposed | API | /api/objects/search/?q=<search> | Returns a list of NEOs as a JSON response \| accepts search parameter |
| Exposed | API | /api/category/<category>/orbits | Returns a list of NEOs as a JSON response \| accepts category search parameter |
| Exposed | API | /api/category/<category> | Returns an object category as a JSON response \| accepts category parameter |
| App-Only | View | /asteroid/random | Returns a random asteroid as a view |
| App-Only | View | /asteroid/<slug> | Returns a detailed asteroid view |
| App-Only | View | /comet/<slug> | Returns a detailed comet view |
| App-Only | View | /asteroid/<slug>/shape | Returns a detailed asteroid view that has an associated DAMIT data |
| App-Only | View | /category/<category> | Returns a list of category view |
| App-Only | View | /solar-system | Returns and populates the solar system visualization tool |
| App-Only | View | /classifications | Returns and populates the solar system visualization tool |

# 5. INTERACTIVE NEO SEARCH ENGINE AND THREE-DIMENSIONAL VISUALIZATION ANALYTICS

## 5.1 Data Visualization Analytics Workflow

The probability of asteroids striking the earth and causing serious damage is very remote, but the devastating consequences of such an impact suggest that we should closely study asteroids to understand their composition, structures, sizes, physical traits, and future trajectories. By doing so, it is possible to make intelligent mitigation plans ahead of time. One way in which this can be done is by leveraging 3D visualization tools that allow users to view asteroids orbital path, shape models, raw materials, and so on. With the growing popularity of the WebGL standards, several open-source 3D engines based on HTML5 have emerged. Three.js is one of the most prevalent engines, owing to its simplicity of use and broad 3D file format compatibility. In this section, we show how open-source libraries, such as Three.js (Dirksen 2013) and Spacekit.js (Ian Webster n.d.), are used to power the Planetary Defense Knowledge Gateway's 3D visualization tool. This tool is integrated with the search engine, which allows users to visualize over

1,400,000 indexed celestial objects. Figure 17 shows the technical architecture of the visualization tool research.

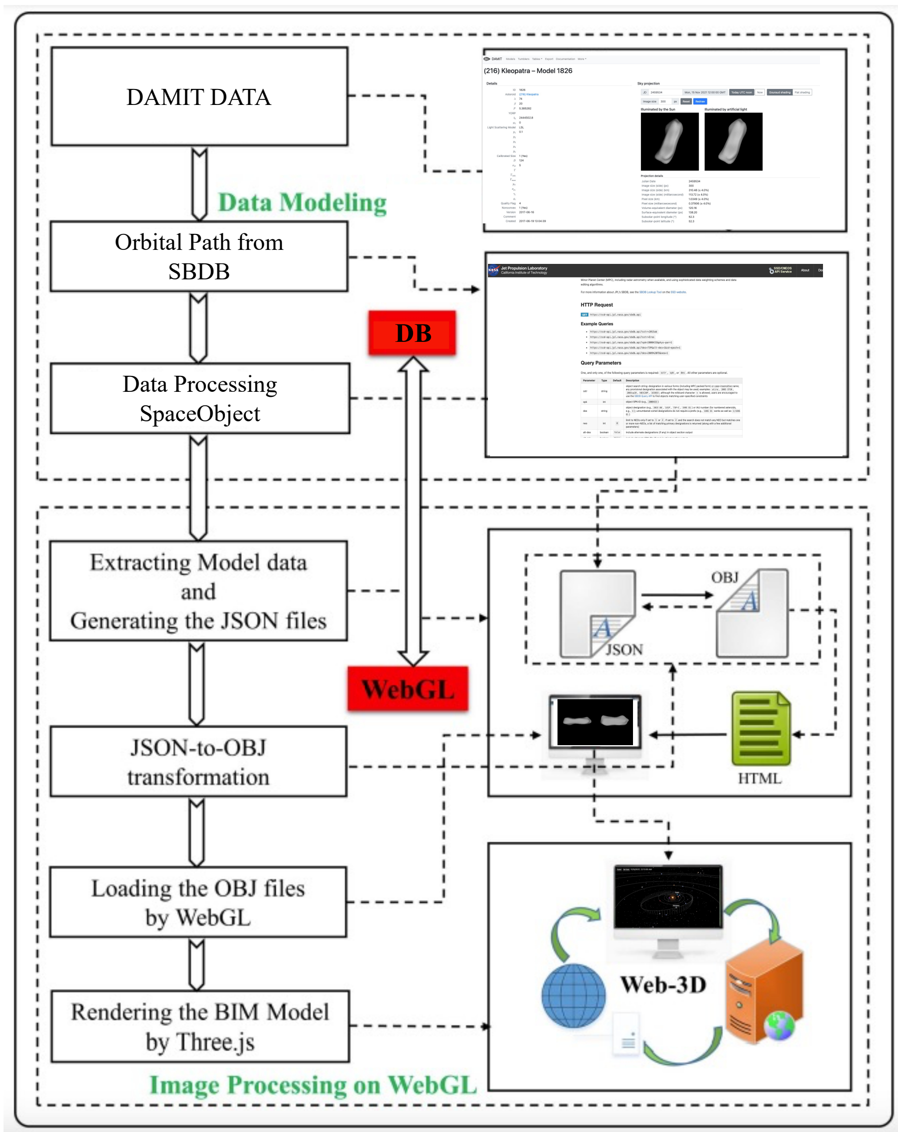



**Figure 17: Visualization Tool Technical Route**

Three.js is an open-source JavaScript library that makes it easy to create 3D animations and interactive graphics in the browser. It provides a wide range of 3D-related features, including support for powerful WebGL rendering, physics, audio, image loading, scene graph management, network communications, and much more. Three.js has extensive cross-browser compatibility and is available for both Firefox and Chrome browsers on desktop platforms. In addition, it can also be used to render asteroids in mobile browsers. Using Three.js reduces workload compared to developing custom visualization tools from scratch, as it takes care of the graphics rendering and provides a high degree of flexibility.

SpaceKit.js is a JavaScript library that provides an easy way of creating and managing scenes consisting of multiple bodies within the browser environment. This library generalizes work that is currently used on Asterank, Meteor Showers, and Ancient Earth. It comes with several features such as collision detection capabilities for planetary defense simulations. In addition, SpaceKit gives users access to several simulation-ready asteroids that can be used for detailed studies.

### 5.2 Data Transition Workflow

This research adopts Promise and fetch technology for data transmission. A Promise is a proxy for a value that may be available now, or in the future. Promises are used in Node.js and JavaScript to add asynchronous capabilities to code. Fetch is a JavaScript library that can be used to asynchronously access resources as an alternative to XMLHttpRequest, including the browser's IndexedDB and Web SQL databases. The data is transmitted in JSON format. The code for file request using these two methods is shown in Figure 18.

```
const loadOptions = (inputValue, callback) => {
  return new Promise(resolve => {
    if (inputValue.length < INPUT_LENGTH_MIN) {
      resolve([]);
      return;
    }
    fetch(`/api/objects/search?q=${inputValue}`).then(resp => {
      return resp.json()
    }).then(respJson => {
      resolve(respJson.results.map(result => {
        return {
          label: result.fullname,
          vizLabel: result.name,
          value: result.slug,
          ephem: result.ephem,
        };
      }));
    });
  });
};
```

```
addMany(objs, opts) {
  opts = opts || {};
  objs.forEach(obj => {
    const key = `spaceobject${this._objCount++}`;
    window.viz.createObject(key, Object.assign(window.VIZ_OBJECT_OPTS, {
      ephem: new Spacekit.Ephem(obj.ephem, 'deg'),
      // Show short name
      labelText: opts.showLabel ? obj.vizLabel : undefined,
      hideOrbit: !opts.showOrbit,
      particleSize: opts.particleSize,
      ecliptic: {
        displayLines: opts.displayEclipticLines,
      },
    }));
  });
```

**Figure 18: Data Transfer Model**

## 5.3 Interaction Design

In this dissertation, the visualization tool mainly uses mouse and keyboard interactions. Mouse and keyboard interactions are done through event listeners that are added to any buttons or other widgets. Mouse interactions can manipulate the position of objects, as well as control camera movements. Keyboard inputs can be used for scrolling, zooming in and out, rotating the camera, and panning around. There are also several features that can be accessed through configurable keyboard shortcuts: change distance to asteroids, adjust asteroids size and color and change asteroids orbit parameters.

```
function handleKeyDown( event ) {

    // console.log( 'handleKeyDown' );

    var needsUpdate = false;

    switch ( event.keyCode ) {

        case scope.keys.UP:
            pan( 0, scope.keyPanSpeed );
            needsUpdate = true;
            break;

        case scope.keys.BOTTOM:
            pan( 0, - scope.keyPanSpeed );
            needsUpdate = true;
            break;

        case scope.keys.LEFT:
            pan( scope.keyPanSpeed, 0 );
            needsUpdate = true;
            break;

        case scope.keys.RIGHT:
            pan( - scope.keyPanSpeed, 0 );
            needsUpdate = true;
            break;

    }
```

**Figure 19: Interaction design model**

### 5.4 Memory Optimization Approach

When an asteroid model is displayed on the web, the browser becomes the rendering container. The maximum amount of memory that may be utilized by the browser is rather limited. When the system's tolerance is exceeded, the browser will nudge the user to restart the computer to avoid it from crashing. Aiming to address the forementioned difficulty, this dissertation utilizes the *Clone* approach. When an item is cloned, only one copy of the cloned object is kept in memory, and the cloned object does not require any additional space. The number of objects in the scene is used to determine which clone should be displayed. The number of reused models accounts for around 80-90 percent of the total number of asteroid models rendered at once. The memory consumption may be considerably reduced when the *Clone Method* is utilized. Furthermore, we can reduce memory usage even further by utilizing Three.js's built-in *Dispose()* function. When we tried several variations, we discovered that when the model is dynamically loaded, the memory will cease expanding. As a result, it may guarantee that adequate memory is used. This dissertation also employs LPM technology to break down and simplify the model, and then sends it to the front-end for preliminary rendering. Procedures such as those may be utilized to assure that the form of the model is unaltered, and that the web page's typical rendering occurs.

### 5.5 GPU Optimization Approach

*Drawcall* is the method that CPU uses to call graphics programming, and it is used by GPU to execute the rendering operation. The browser will carry out one frame's worth of rendering each time. When the operation of *Drawcall* becomes more difficult or there are a large number of vertices in the scene, the *Drawcall* count will rise, and rendering time will grow, lowering FPS. The above issue may be addressed by applying the Three.js' *Merge* approach. The *Merge* strategy merges numerous distinct geometries into a single form to minimize the number of geometries in the scene. Materials must also be assigned once the geometry is completed. The materials are different depending on the object type, but models of the same type have the same material. As a result, we may combine all models of the same material into one geometry and generate the mesh when merging. Therefore, the number of model objects in the scene will drop from tens of thousands to just a few, resulting in significant reduction on Drawcalls.

### 5.6 Results and Experiments

There are several tests and experiments that were conducted to determine if there is any significant difference in terms of data handling and benchmark performance. In terms of data handling capability testing, three visualization tools were considered – the planetary defense mitigation gateway (PDKG), CNEOs JPL, and Asterank. The first test was a frames-per-second test, which is a measure of how fast a tool can render images. The second test was a data handling experiment, which is a measure of how well a browser can handle data requests without causing significant frames-per-second fluctuation or resulting in a browser crash. The frames-per-second test was conducted by averaging the frame rate of three consecutive tests. In Figure 20, the PDKG tool is shown to have a significantly higher frame rate than the other two tools. In terms of data handling, the PDKG tool is able to handle data requests at a rate that is orders of magnitude higher than the other two tools.

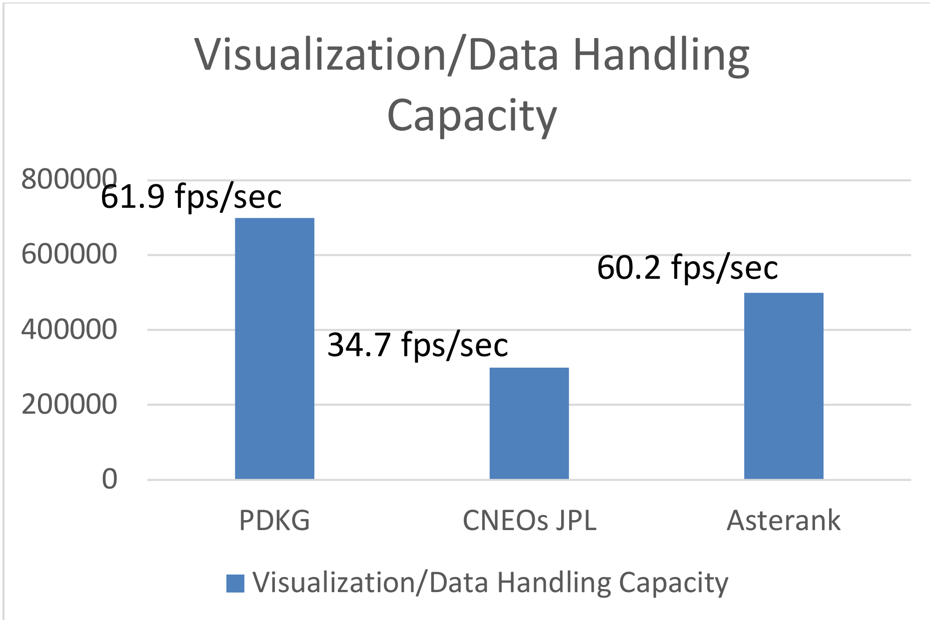



**Figure 20: Visualization/Data Handling Capacity**

The second set of experiments was conducted using the lighthouse score, selenium test, ease of access, and the total number of features comparison. For this set of tests, six visualization tools were considered - planetary defense mitigation gateway (PDKG), CNEOs JPL, Asterank, NASA Eyes, Worldwide Telescope, and the Kerbal Space Program. The lighthouse score is a measure of how well a tool can perform in terms of speed, SEO, and accessibility. The selenium test is a measure of how well a tool can handle dynamic content. The ease of access score is a measure of how easy it is for users to find and use the features of a tool. The total number of features score is a measure of the total number of functionalities provided by each of the visualization tools. Figure 21 shows the results of the lighthouse score experiment, with the PDKG tool having a significantly higher score than the other two tools. In terms of the selenium test, the PDKG tool was able to handle dynamic content at a rate that is orders of magnitude higher than all the tools tested except Asterank. In terms of the total number of features, the PDKG tool is significantly higher than all the other tools, and comparable to Asterank with a total of 17 features. The Asterank tool has a total of 19 features, while the CNEOs JPL and NASA Eyes tools have a total of 14 and 8 features respectively.

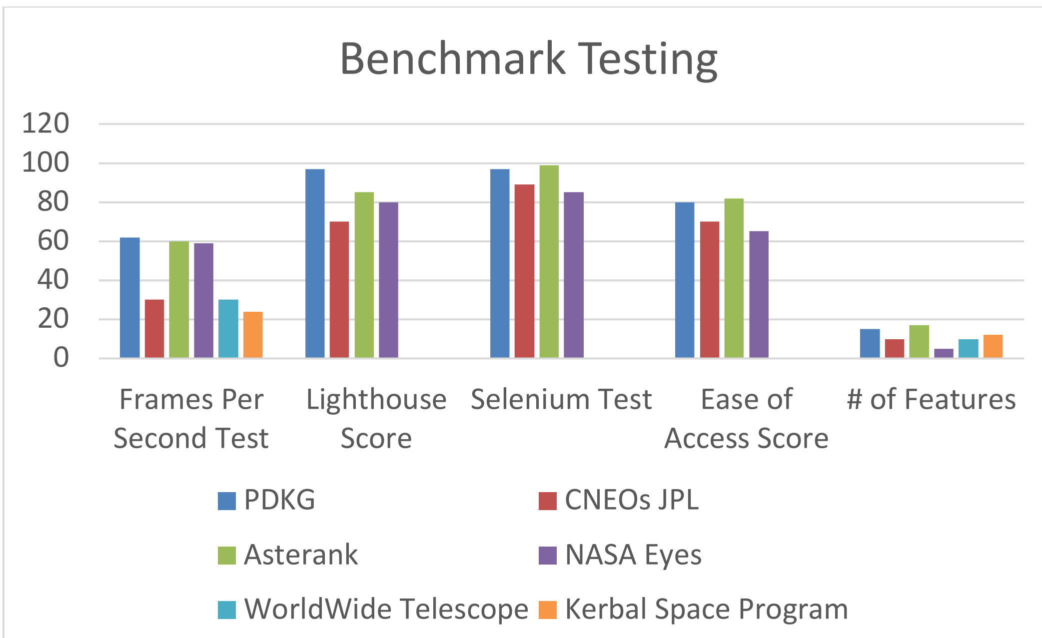



**Figure 21: Benchmark Testing**

In conclusion, the planetary defense mitigation gateway (PDKG) visualization tool is superior to other visualization tools in terms of performance, data handling capability, SEO score, ease of access score, and the total number of features provided.

The planetary defense mitigation gateway (PDKG) is a powerful visualization tool that can be used to visualize data sets of any size and complexity.

# 6. SYSTEM INTRODUCTION

Planetary Defense Knowledge Gateway provides an easy-to-use interface where users can query for over 1,400,000 celestial object datasets, explore random objects, review search results, view categories based on orbital classifications defined by NASA PDS, and interact with several types of visualization tools.

## 6.1 Data Discovery Queries

The large volumes of structured data currently available, from various organizations, such as NASA, MPC, and ESA, to open data portals data, present new opportunities for progress in answering many important scientific questions. However, finding relevant data is difficult due to the scattered nature of information available on the web. While search engines have addressed this problem for web documents, there are many new challenges involved with supporting the discovery of domain-specific structured data. In this section, this research demonstrates how the Django REST API-powered PDKG search engine addresses some of these challenges by providing a user interface that enables users to explore metadata about datasets using celestial object keywords.

Over the last few years, scientists and governments have been publishing datasets that were previously hidden from public access. This move towards transparency has opened opportunities for answering questions of all kinds in an increasingly data-driven world where knowledge is crucial to progress (Castelo et al. 2021). The availability of these scientific observations provides new insights into many important planetary defense research questions issues, including closeness to the Earth. Finding useful data may be difficult, as data are scattered across many sources and repositories. Several techniques have been developed to organize and index data collections to tackle this issue: From specialized repositories like NASA Open Data (NASA Open Data Portal n.d.), which gathers data from other NASA archives, to the Minor Planet Center Data Portal (IAU Minor Planet Center n.d.). While these are a big step toward making data discovery easier, they have one major drawback: They do not provide a solid method for connecting data from various genuine sources with common formats. This severely restricts a user's ability to communicate information demands.

These queries are enabled, in part, by a Django REST API-powered framework that we developed to extract useful information from the actual datasets. This includes not only the unique ID assigned by the Minor Planet Center, but also their slug name identified by NASA. Users can explore large dataset collections through an easy-to-use interface that guides them in the process of specifying complex queries. In the following sections, we give an overview of the user interface and features of the gateway and discuss a few use cases that we will present during our demonstration. Users will be able to interact directly with the PDMG to query over 1,400,000 datasets.

Users can query the indices by specifying keywords (see Figure 19(a)). Users search for datasets by specifying the celestial object name, unique ID, or the slug name defined in the Small-Body Database (SBDB). To see search results, users would have to select a filtered option from the dynamic dropdown (see Figure 17). Inspired by Google Search Engine's "Feeling Lucky" feature, this application also provides similar functionality to view a random object (see Figure 22(b)). The header navigation can be found on the top right corner of the gateway's home page. The navigation bar has quick access to the Search, and the Categories page (see Figure 22(c)).

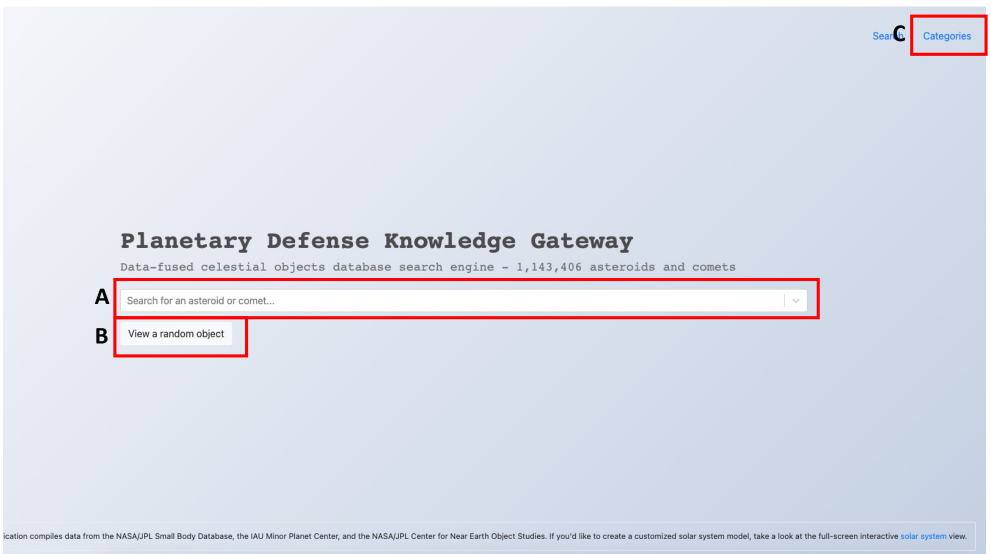



**Figure 22: Components of the PDMG's user interface: search input field (a); view a random object (b); Navigation bar – Categories (c)**

### 6.2 Search Result

Datasets, unlike web documents that may be summarized with short meta descriptions, have many components to consider when assessing their relevance. As a result, search results must be presented in an easy-to-understand format. Figure 23 shows the search results when the term 'PF184' is entered. On the top, there is a container that identifies the size and classifies the type of asteroid being displayed on the screen (see Figure 21(a)). A two-dimensional representation of the asteroid's orbit path relative to the major planets is shown in that same container, located on the right side. Based on the asteroid's physical characteristics and orbital elements, several key facts are computed on the client side. Those computations then populate the Key Facts, Similar Objects & References, and Map Comparison containers (see Figure 23(c,d,e)). The orbital elements are used to determine the category of the asteroid that was searched. Furthermore, the Sentry-provided merged data are utilized to identify whether the object can be hazardous or not. The diameter and the magnitude of the asteroid are used to compute the object's comparable size. If the asteroid has comparable orbits to other asteroids in the database, three objects with similar paths are displayed in the Similar Objects area. The References section provides dynamic links to access the object's external information sources. The Map Comparison component loads the map with Google Maps JavaScript API and draws the impact circle based on the object's physical characteristics. The Orbital Elements, Physical Characteristics, and Derived Characteristics are all retrieved from the database without using any DOM manipulations.

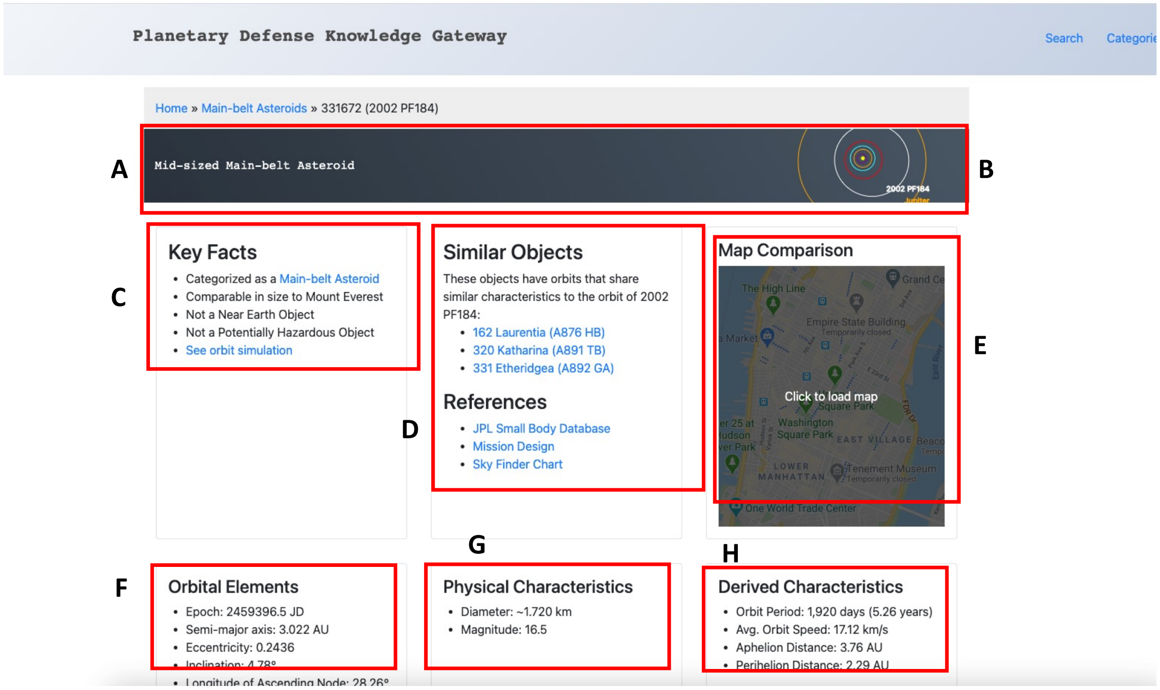



**Figure 23: Components of the Searched Result: Classification of the asteroid (a) and orbit diagram(b); key facts (c); similar objects and references (d); map comparison (e); orbital elements(f), physical characteristics (g); and derived characteristics (h)**

## 6.3 View Category

The Categories page can be divided into two parts: Asteroids with known shapes, and categories with orbital classifications defined by NASA Planetary Data System. Asteroids with known shapes are those that have been derived using the *Light-Curve Inversion* method, combined with other inversion techniques in some cases. The system currently possesses 1,609 asteroids with known shapes, accounting for 0.1% of objects. Table 11 is a list of categories and their descriptions that have been established by NASA Planetary Data System (PDS). The categories link may be utilized to navigate the user to a more detailed category view (see Figure 24(a,b)).

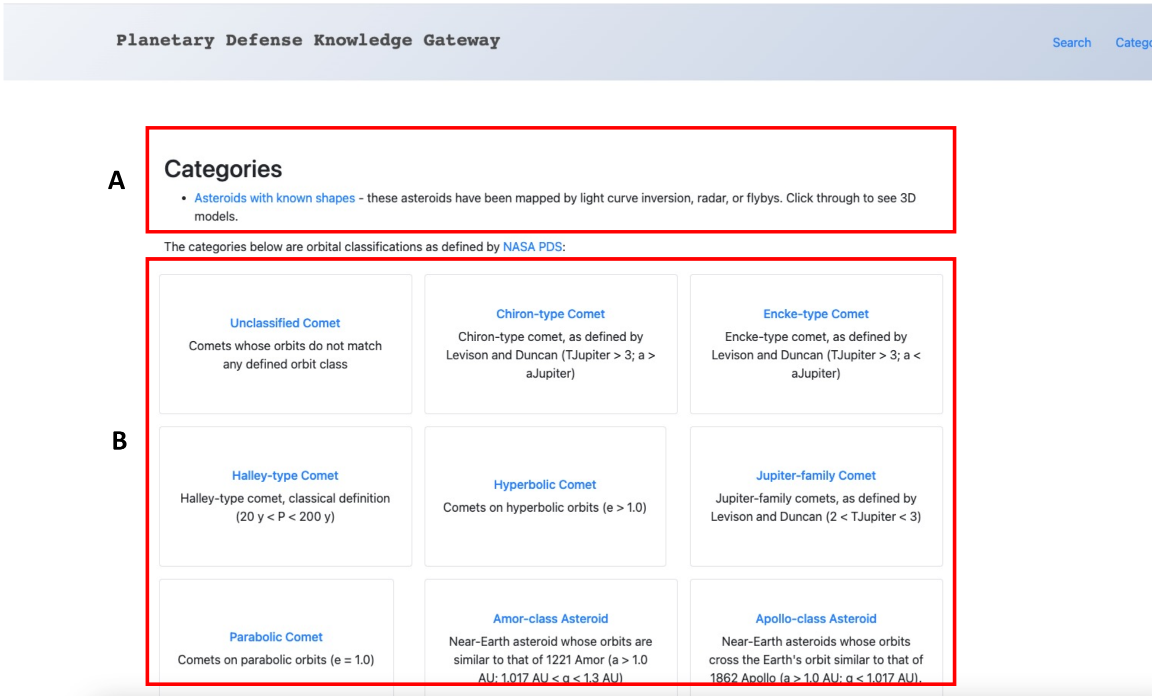



**Figure 24: Asteroids with known shapes (a); categories defined by the NASA Planetary Data System (b)**

**Table 11: List of categories and descriptions established by NASA PDS**

| Orbital Classification | Description |
|---|---|
| Unclassified Comet | Comets whose orbits do not match any defined orbit class |
| Chiron-type Comet | Chiron-type comet, as defined by Levison and Duncan (TJupiter > 3; a > aJupiter) |
| Encke-type Comet | Encke-type comet, as defined by Levison and Duncan (TJupiter > 3; a < aJupiter) |
| Halley-type Comet | Halley-type comet, classical definition (20 y < P < 200 y) |
| Hyperbolic Comet | Comets on hyperbolic orbits (e > 1.0) |
| Jupiter-family Comet | Jupiter-family comets, as defined by Levison and Duncan (2 < TJupiter < 3) |
| Parabolic Comet | Comets on parabolic orbits (e = 1.0) |
| Amor-class Asteroid | Near-Earth asteroid whose orbits are similar to that of 1221 Amor (a > 1.0 AU; 1.017 AU < q < 1.3 AU) |
| Apollo-class Asteroid | Near-Earth asteroids whose orbits cross the Earth's orbit similar to that of 1862 Apollo (a > 1.0 AU; q < 1.017 AU). |
| Asteroid | Asteroid orbit not matching any defined orbit class |
| Aten-class Asteroid | Near-Earth asteroid orbits similar to that of 2062 Aten (a < 1.0 AU; Q > 0.983 AU) |
| Centaur-class Asteroid | Objects with orbits between Jupiter and Neptune (5.5 AU < a < 30.1 AU) |
| Hyperbolic Asteroid | Asteroids on hyperbolic orbits (e > 1.0) |
| Interior-Earth Asteroid | Asteroids with orbits contained entirely within the orbit of the Earth (Q < 0.983 AU) |
| Inner Main-belt Asteroid | Asteroids with orbital elements constrained by (a < 2.0 AU; q > 1.666 AU) |
| Main-belt Asteroid | Asteroids with orbital elements constrained by (2.0 AU < a < 3.2 AU; q > 1.666 AU) |
| Mars-crossing Asteroid | Asteroids that cross the orbit of Mars constrained by (1.3 AU < q < 1.666 AU; a < 3.2 AU) |
| Outer Main-belt Asteroid | Asteroids with orbital elements constrained by (3.2 AU < a < 4.6 AU) |
| Parabolic Asteroid | Asteroids on parabolic orbits (e = 1.0) |
| Jupiter Trojan | Asteroids trapped in Jupiter's L4/L5 Lagrange points (4.6 AU < a < 5.5 AU; e < 0.3) |
| Trans-Neptunian Object | Objects with orbits outside Neptune (a > 30.1 AU) |

## 6.4 Detailed Category View and Orbit Explorer

The following page (see Figure 25) also replicates the look of the search result page, with the top container that resembles that in Figure 22(a). The only difference can be seen in Figure 25(a), where the system returns the total percentage of objects matching the same object classification. For example, when *Asteroids with Known Shapes* are selected, then the string returns: "There are 1,609 asteroids with known shapes of this type in the database out of 1,143,406 total, accounting for 0.1% of objects." A filtered search container can be found on the left side of the page (see Figure 25(b)). The objects listed on that container are linked with the orbit explorer visualization tool (see Figure 25(c)). Hovering over a specific object triggers the orbit explorer to also highlight the orbital path of the object. There are a few built-in capabilities on the orbit explorer to choose a particular day or speed up/slow down how quickly time passes in the tool. One can also expand the orbit explorer to full screen, add additional objects to the explorer view for further analysis (see Figure 26).

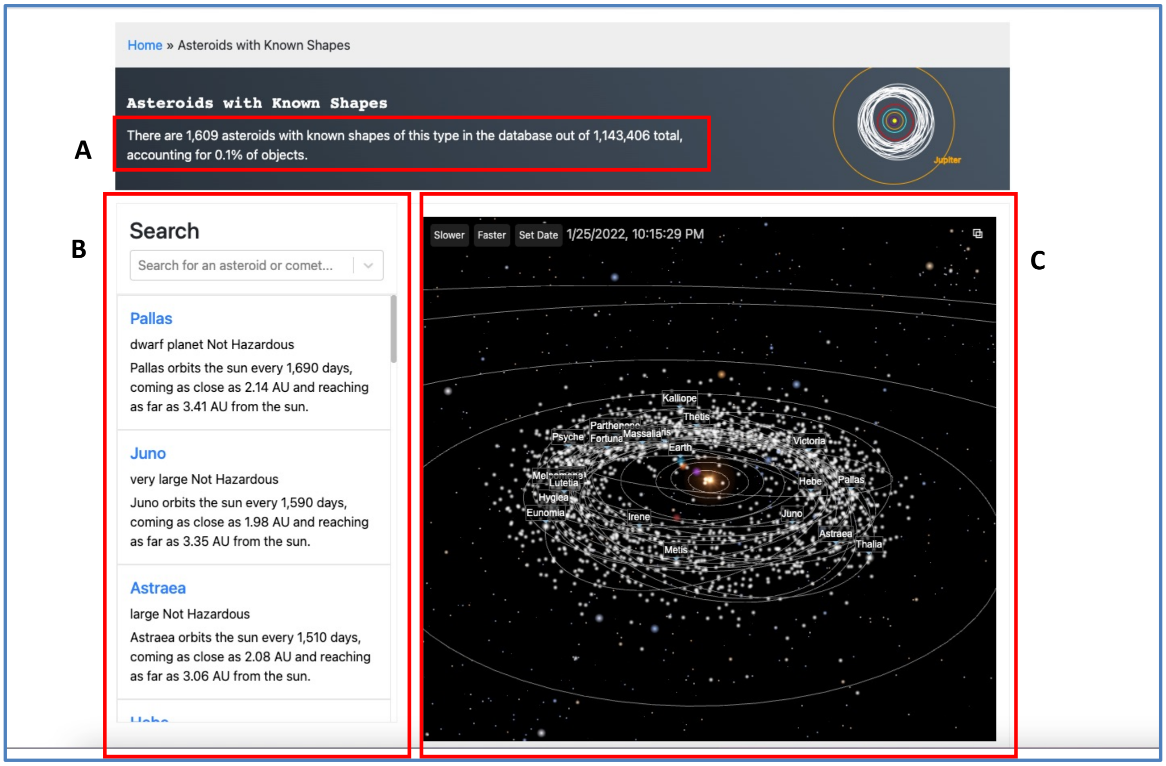



**Figure 25: Classified objects / total objects in percentage (a); search container (b); orbit explorer (c)**

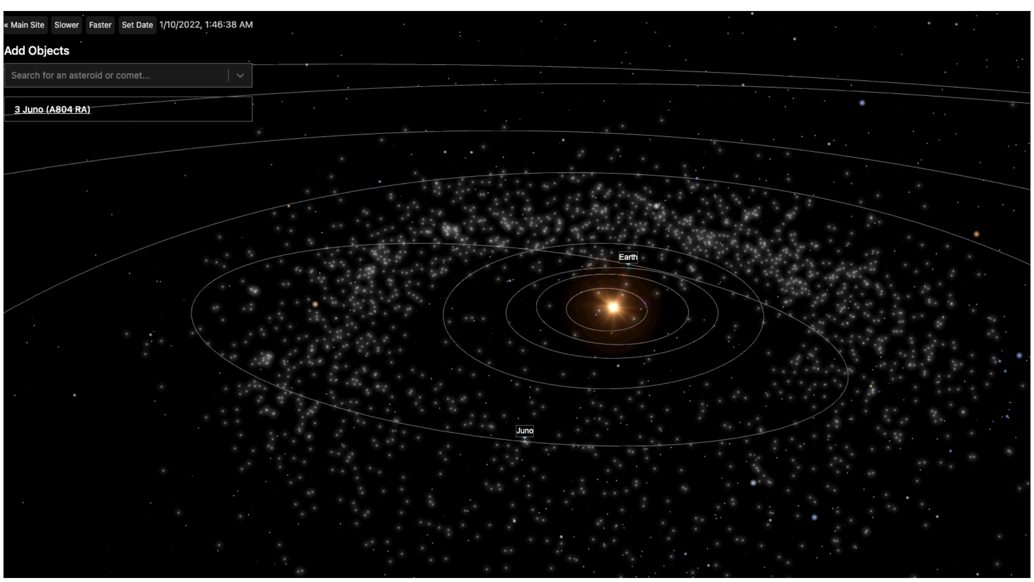
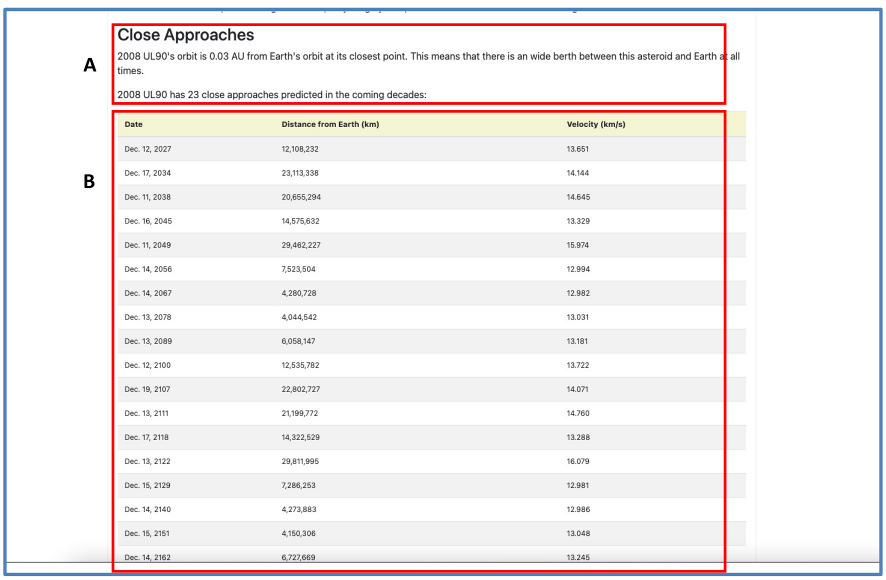



**Figure 26: Full-Screen Orbit Explorer**

## 6.5 Close-Approach

A close approach is defined as an asteroid within 0.05 AU from Earth on a trajectory that will bring it closer than 0.05 AU to our planet in the next couple of years. The close approach table can be found in the search results screen. If an object is labeled as a near-Earth object or potentially hazardous, the gateway pulls data from the database's close-approach table. The system iterates through the close-approach table and calculates the total number of close approaches predicted in the coming decades (see Figure 27(a). It also displays a table that lists out the date, distance from Earth in (km), and Velocity (see Figure 27(b)). This information can be used to predict the future path of an asteroid.




| Date | Distance from Earth (km) | Velocity (km/s) |
|---|---|---|
| Dec. 12, 2027 | 12,108,232 | 13.651 |
| Dec. 17, 2034 | 23,113,338 | 14.144 |
| Dec. 11, 2038 | 20,655,294 | 14.645 |
| Dec. 16, 2045 | 14,575,632 | 13.329 |
| Dec. 11, 2049 | 29,462,227 | 15.974 |
| Dec. 14, 2056 | 7,523,504 | 12.994 |
| Dec. 14, 2067 | 4,280,728 | 12.982 |
| Dec. 13, 2078 | 4,044,542 | 13.031 |
| Dec. 13, 2089 | 6,058,147 | 13.181 |
| Dec. 12, 2100 | 12,535,782 | 13.722 |
| Dec. 19, 2107 | 22,802,727 | 14.071 |
| Dec. 13, 2111 | 21,199,772 | 14.760 |
| Dec. 17, 2118 | 14,322,529 | 13.288 |
| Dec. 13, 2122 | 29,811,995 | 16.079 |
| Dec. 15, 2129 | 7,286,253 | 12.981 |
| Dec. 14, 2140 | 4,273,883 | 12.986 |
| Dec. 15, 2151 | 4,150,306 | 13.048 |
| Dec. 14, 2162 | 6,727,669 | 13.245 |

**Figure 27: N number of close approaches predicted in the coming decades (a); close-approach table (b)**

## 6.6 Shape Model and Artistic Rendering

There are currently 1,609 asteroids with known shapes models in the system. These shape models can be found on the search result page. If the gateway identifies that the database contains a shape model for the selected object, then the system automatically appends the rendered image of the model to the page (see Figure 28(a). Users can also view and interact with the shape model in 3D (see Figure 28(b). For example, a rendered shape model for Juno is shown in Figure 29.

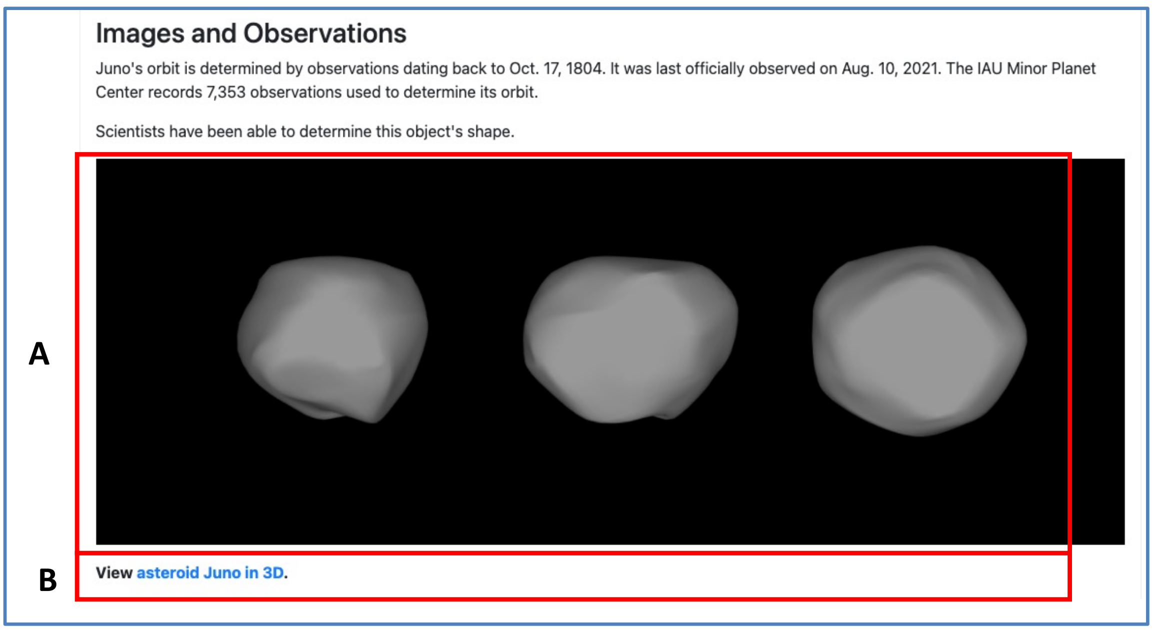



**Figure 28: Juno's rendered image (a); link to view interactive 3D model of Juno (b)**

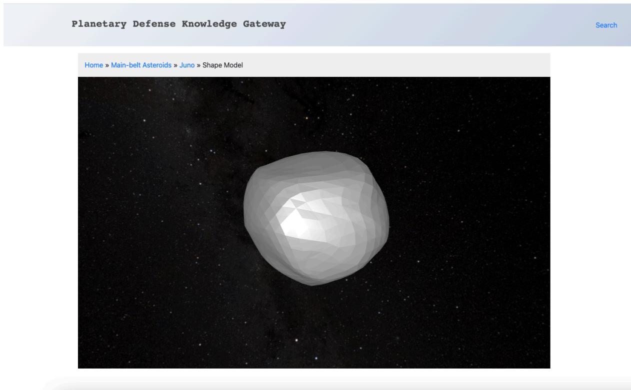



**Figure 29: Interactive 3D view of Juno**

The application also compares the size of objects against approximate Earth landscapes. For example, In Figure 30, an artistic rendering of Hathor is contrasted against a rough landscape rendering of New York City in the backdrop. This approximation is intended for full-resolution desktop browsers. The asteroid's form, color, and texture are imaginary.


**Figure 30: Artistic Rendering of Hathor**

# 7. CONTRIBUTIONS AND CONCLUSION

## 7.1 Contributions

To recall, the primary challenge that this dissertation addressed is that information on detecting, characterizing, and mitigating NEO threats is presently scattered. There is a lack of structured architecture, integration, and interoperability in the planetary defense domain. The solution proposed in this dissertation is an interoperable framework for planetary defense data integration and visualization. This framework can be used to connect disparate pieces of information, support coordinated observations, verify observations (or extensions of physics-based models), and produce appropriate visualizations.

The approach taken in this dissertation was to first develop a scalable data-fusion framework to integrate dispersed and diverse information residing at different organizations. Although this framework can incorporate data sources from any organization that provides their data sets in a format that can be ingested by the framework, such as. JSON, .CSV, ASCII formats. For this dissertation, we focused on data sources that are particularly popular in the planetary defense domain. The data fusion framework was designed to overcome many of the challenges associated with integrating information from these disparate sources, including the lack of a common ontology, the use of different standards and formats, and the variety of access mechanisms.

To resolve the issue of inefficient processes and decisions made on incomplete data, this dissertation also focused on multiprocessing techniques, comprehensive data modeling, and data inaccuracies verification. The multiprocessing techniques consisted of three main techniques: (1) a data pre-fetching technique to minimize data retrieval latency, (2) an in-memory caching technique to improve data access performance, and (3) a query parallelization technique to speed up the execution of complex queries. The comprehensive data modeling considered the different types of information that needed to be integrated, such as observational data, catalog data, and expert knowledge. The data inaccuracies verification was performed using a set of heuristics that were designed to identify errors in the data. The data inaccuracies verification was performed using a set of heuristics that were designed to identify errors in the data. The heuristics were based on the following three principles: (1) redundant data, (2) contradictory data, and (3) anomalous data.

This dissertation also identified challenges related to discrepancies between PD data formats. To resolve discrepancies between PD data formats, this dissertation proposed a four-step data pipeline. The first step is to detect the discrepancies between the PD data formats. The second step is to automatically generate transformations to map the detected discrepancies to a common format. The third step is to automatically verify the generated transformations. The fourth and final step is to load the data into a common format that can be used by the Planetary Defense Knowledge Gateway (PDKG).

We then demonstrated how the data-fusion framework could be used to develop the Planetary Defense Knowledge Gateway (PDKG). The PDKG is a platform that enables users to access, visualize, and analyze integrated, and interoperable planetary defense data. The PDKG was designed to address the needs of three different user groups: (1) planetary defense experts, (2) planetary defense decision-makers, and (3) the general public. The PDKG was developed using web-based technologies and standards so that it can be easily deployed and accessed by users from any location with an internet connection. Furthermore, the PDKG incorporates a modular design so that new functionality can be added as needed. For example, if a new planetary defense data source becomes available, it can be added to the PDKG by simply adding a new module to the system. The next step was to incorporate a set of visualization tools that could be used to explore and analyze the integrated planetary defense information. This set was designed to support both scientific analysis and public outreach. To facilitate scientific analysis, the tool provides a variety of ways to filter, aggregate, and visualize the data. To support public outreach, the visualization tool includes a set of predefined views that show the most important information in an easily understandable way. Table 12 shows a summary of problems and solutions that were addressed in this dissertation.

**Table 12: Problems and Contributions addressed in this dissertation**

| **Problems** | **Contributions** |
|---|---|
| Heterogeneity of the situation<br>Data are **dispersed** throughout different organizations (Yang et al., 2017) | (3) Developed a data-fusion framework to integrate dispersed and diverse information residing at different organizations. |
| Lack of structured architecture, integration, and interoperability | (4) Multiprocessing techniques have been applied to process large datasets that were utilized in the system. |

| | |
|---|---|
| - Resulting in **inefficient processes** and decision made on **incomplete data** (Wu et al., 2014) | (3.5) Comprehensive Data Model<br>(4.3) Data Inaccuracies verification |
| Diversity of data and information for PD research<br>- Discrepancy between **PD data formats** and lack of visibility (Yang et al., 2017)<br>- Lack of collaboration and data fusion among different data sources (Yang et al., 2019) | (3.4) Data pipelines have been developed with loosely-coupled architecture in mind. This will allow future integration to be applied seamlessly.<br>(6) Designed an API-driven keyword-centric NEOs search engine that allows users to search from 1,143,406 celestial objects. |
| Future threats mitigation<br>Lack of **visualization technologies** that offer interactive simulation (Shams et al. 2019) | (5) Several visualization tools have been provided to analyze Near-Earth approaches in a three-dimensional environment. |

### 7.2 Conclusion

The Planetary Defense Knowledge Gateway framework involves the integration of data, information, and knowledge from a wide variety of agencies and organizations throughout the world. Efficient integration of such resources in a seamless fashion, and its communication among stakeholders, will be critical for mitigating a potential threat to our planet by a near-Earth object. This dissertation has presented a data-fusion framework that can be used to support the detection, characterization, and mitigation of potentially hazardous asteroids. The data-fusion framework was used to develop the Planetary Defense Knowledge Gateway (PDKG), a platform that enables users to access, visualize, and analyze integrated, and interoperable planetary defense data. The PDKG was designed to address the needs of three different user groups: (1) planetary defense experts, (2) planetary defense decision-makers, and (3) the general public. The PDKG was developed using web-based technologies and standards so that it can be easily deployed and accessed by users from any location with an internet connection. Furthermore, the PDKG incorporates a modular design so that new functionality can be added as needed. The data-fusion framework and planetary defense knowledge gateway are effective for supporting planetary defense activities. The data-fusion framework is scalable and can be used to support planetary defense activities.

### 7.3 Solution Impact

The data-fusion framework and planetary defense knowledge gateway are valuable assets that can be used to protect Earth from potentially hazardous asteroids. If the PD-related communities, agencies, and/or organizations adapt the system, then they can use the PDKG to make better-informed decisions about planetary defense mitigation strategies. In addition, the data-fusion framework and planetary defense knowledge gateway can be used to support public outreach efforts by providing the public with easy access to planetary defense information. Which also can help raise awareness about the importance of planetary defense. The PDKG can also act as a central repository for API discovery and open-source research sharing if properly leveraged by the PD communities. Furthermore, the system provides a mechanism for public participation in dynamic visualization studies. The planetary defense knowledge gateway has the potential to significantly improve the way planetary defense data is managed and used.

### 7.4 Future Research

The planetary defense domain is an evolving field with new data sources, methods, and technologies being developed all the time. The PDKG has the potential to expand future research in several areas. For example, automatic semantic annotation of planetary defense data sources can be used to improve the usability of the PDKG. In addition, new data sources can be integrated into the system as they become available. Furthermore, the visualization tool can be enhanced to support more advanced analysis and public outreach efforts. Finally, the PDKG can be extended to support other planetary defense activities such as planetary hazard identification, risk assessment, and mitigation.

To keep up with the latest advances, the PDKG must be updated on a regular basis with fresh information. To support this continuous update process, the PDKG will need to deploy a set of automated tasks that are designed to keep the system up-to-date with the latest planetary defense data. These tasks include fetching new data from planetary defense data sources, processing the data, and updating the PDKG database. Future research will aim to incorporate a dynamic mitigation scenario simulation based on rocket trajectory and deflection variables into the visualization tool. Currently, our tool only supports 4D (x, y, z, and time variables), and the "uncertainty" variable is absent from the system. The research will concentrate on incorporating the trajectory's unpredictability.

Future research will also involve improving the web crawler, enhancing search performance, and inviting domain experts to evaluate the findings concerning the interactive smart search module. By applying the structure as is, additional research will include adding data from other sources after minor modifications are made. This will allow the program to provide further information that may be useful to researchers. Latent semantic analysis techniques will be used to enhance our planet defense knowledge base.

It will also be beneficial to evaluate the effectiveness of the data-fusion framework and planetary defense knowledge gateway. The evaluation shall consist of two parts: (1) a user study and (2) a performance assessment. The user study will be conducted with planetary defense experts from NASA, JPL, and other institutions. The performance assessment will be conducted to assess the scalability of the data-fusion framework. In summary, the research will be continued to push and develop the Planetary Defense community's knowledge while also organizing the complicated system to improve efficiency for emergency response.

## REFERENCES


Accessible NEAs. n.d. "Accessible NEAs." Retrieved November 9, 2020 (https://www.minorplanetcenter.net/iau/MPDes.html).

Alchin, Marty, Jacob Kaplan-Moss, and George Vilches. 2013. *Pro Django*. Vol. 2. Springer.

Anon. n.d. "Celestia: Home." Retrieved February 7, 2019 (https://celestia.space/).

Anon. n.d. "Kerbal Space Program - Create and Manage Your Own Space Program." Retrieved February 7, 2019 (https://www.kerbalspaceprogram.com/).

Anon. n.d. "NASA's Eyes." Retrieved February 7, 2019 (https://eyes.nasa.gov/).

Anon. n.d. "NEO Basics." Retrieved February 28, 2019 (https://cneos.jpl.nasa.gov/about/basics.html).

Anon. n.d. "Universe Sandbox." Retrieved February 7, 2019 (http://universesandbox.com/).

Anon. n.d. "WorldWide Telescope Web Client." Retrieved February 7, 2019 (http://www.worldwidetelescope.org/webclient/).

Arentz, Robert, Harold Reitsema, Jeffrey Van Cleve, and Roger Linfield. 2010. "NEO Survey: An Efficient Search for Near-Earth Objects by an IR Observatory in a Venus-like Orbit." Pp. 418–29 in *AIP Conference Proceedings*. Vol. 1208. American Institute of Physics.

Barbee, Brent, Ronald Mink, and Daniel Adamo. 2011. "Methodology and Results of the Near-Earth Object (NEO) Human Space Flight (HSF) Accessible Targets Study (NHATS)." Girdwood, AK.

Barbee, Brent W. 2014. "Near-Earth Asteroids: Destinations for Human Exploration." Baltimore, MD.

Board, Space Studies, and National Research Council. 2010. *Defending Planet Earth: Near-Earth-Object Surveys and Hazard Mitigation Strategies*. National Academies Press.

Boattini, A., G. D'abramo, G. Forti, and R. Gal. 2001. "The Arcetri NEO Precovery Program." *Astronomy & Astrophysics* 375(1):293–307.

Bonilla, Dennis. 2015. “Gravity Tractor.” *NASA*. Retrieved November 8, 2020 (http://www.nasa.gov/content/asteroid-grand-challenge/mitigate/gravity-tractor).

Bornman, Hester, and S. H. von Solms. 1993. “Hypermedia, Multimedia and Hypertext: Definitions and Overview.” *The Electronic Library* 11(4/5):259–68. doi: 10.1108/eb045243.

Bu, Di N. n.d. “Master of Science Computer Science School of Informatics University of Edinburgh 2003.” 75.

Castanedo, Federico. 2013. “A Review of Data Fusion Techniques.” *The Scientific World Journal*. Retrieved August 11, 2020 (https://www.hindawi.com/journals/tswj/2013/704504/).

Castelo, Sonia, Rémi Rampin, Aécio Santos, Aline Bessa, Fernando Chirigati, and Juliana Freire. 2021. “Auctus: A Dataset Search Engine for Data Augmentation.” *ArXiv:2102.05716 [Cs]*.

Chiarenza, Alfio Alessandro, Alexander Farnsworth, Philip D. Mannion, Daniel J. Lunt, Paul J. Valdes, Joanna V. Morgan, and Peter A. Allison. 2020. “Asteroid Impact, Not Volcanism, Caused the End-Cretaceous Dinosaur Extinction.” *Proceedings of the National Academy of Sciences* 117(29):17084–93.

Chodas, Paul. 2015. “Overview of the JPL Center for NEO Studies (CNEOS).” Pp. 214–09 in *AAS/Division for Planetary Sciences Meeting Abstracts# 47*. Vol. 47.

Cox, D. 1972. “Delay Doppler Characteristics of Multipath Propagation at 910 MHz in a Suburban Mobile Radio Environment.” *IEEE Transactions on Antennas and Propagation* 20(5):625–35.

Curry, Edward. 2016. “The Big Data Value Chain: Definitions, Concepts, and Theoretical Approaches.” Pp. 29–37 in *New horizons for a data-driven economy*. Springer, Cham.

Daou, Doris, and Lindley Johnson. 2019. “A Summary of the United States’s National Near-Earth Object Preparedness Strategy and Action Plan.” P. EPSC-DPS2019 in *EPSC-DPS Joint Meeting 2019*. Vol. 2019.

De Angelis, Daniela, Anne M. Presanis, Paul J. Birrell, Gianpaolo Scalia Tomba, and Thomas House. 2015. “Four Key Challenges in Infectious Disease Modelling Using Data from Multiple Sources.” *Epidemics* 10:83–87. doi: 10.1016/j.epidem.2014.09.004.

Dirksen, Jos. 2013. *Learning Three.Js: The JavaScript 3D Library for WebGL*. Packt Publishing Ltd.

Drakopoulos, Georgios, Panagiotis Gourgaris, and Andreas Kanavos. 2018. "Graph Communities in Neo4j: Four Algorithms at Work." *Evolving Systems* 11(3):397–407. doi: 10.1007/s12530-018-9244-x.

Ďurech, J., J. Hanuš, D. Oszkiewicz, and R. Vančo. 2016. "Asteroid Models from the Lowell Photometric Database." *Astronomy & Astrophysics* 587:A48. doi: 10.1051/0004-6361/201527573.

Ďurech, J., V. Sidorin, and M. Kaasalainen. 2010. "DAMIT: A Database of Asteroid Models." *Astronomy & Astrophysics* 513:A46. doi: 10.1051/0004-6361/200912693.

Ďurech, J., V. Sidorin, and Mikko Kaasalainen. 2010. "DAMIT: A Database of Asteroid Models." *Astronomy & Astrophysics* 513:A46.

Emrich, Tobias, Hans-Peter Kriegel, Nikos Mamoulis, Matthias Renz, and Andreas Züfle. 2012. "Indexing Uncertain Spatio-Temporal Data." Pp. 395–404 in *Proceedings of the 21st ACM international conference on Information and knowledge management*, *CIKM '12*. New York, NY, USA: Association for Computing Machinery.

Emrich, Tobias, Hans-Peter Kriegel, Nikos Mamoulis, Matthias Renz, and Andreas Zufle. 2012. "Querying Uncertain Spatio-Temporal Data." Pp. 354–65 in *2012 IEEE 28th International Conference on Data Engineering*.

Farquhar, Robert, Jun'ichiro Kawaguchi, C. Russell, Gerhard Schwehm, Joseph Veverka, and Donald Yeomans. 2002. "Spacecraft Exploration of Asteroids: The 2001 Perspective." *Asteroids III* 367–76.

Franke, Richard, and Gregory M. Nielson. 1991. "Scattered Data Interpolation and Applications: A Tutorial and Survey." *Geometric Modeling* 131–60.

Garner, Rob. 2015. "OSIRIS-REx." *NASA*. Retrieved November 14, 2021 (http://www.nasa.gov/osiris-rex).

GlobalInterpreterLock. n.d. "GlobalInterpreterLock - Python Wiki." Retrieved November 12, 2021 (https://wiki.python.org/moin/GlobalInterpreterLock).

Gustafson, John L. 2011. "Amdahl's Law." Pp. 53–60 in *Encyclopedia of Parallel Computing*, edited by D. Padua. Boston, MA: Springer US.

Hanuš, J., J. Ďurech, M. Brož, Brian D. Warner, Frederick Pilcher, Robert Stephens, Julian Oey, Laurent Bernasconi, Silvano Casulli, and Raoul Behrend. 2011. "A Study of Asteroid Pole-Latitude Distribution Based on an Extended Set of Shape

Models Derived by the Lightcurve Inversion Method." *Astronomy & Astrophysics* 530:A134.

Harris, A. W., J. W. Young, L. Contreiras, T. Dockweiler, L. Belkora, H. Salo, W. D. Harris, E. Bowell, M. Poutanen, and R. P. Binzel. 1989. "Phase Relations of High Albedo Asteroids: The Unusual Opposition Brightening of 44 Nysa and 64 Angelina." *Icarus* 81(2):365–74.

Holovaty, Adrian, and Jacob Kaplan-Moss. 2009. *The Definitive Guide to Django: Web Development Done Right*. Apress.

Ian Webster. n.d. "Spacekit." Retrieved November 14, 2021 (https://typpo.github.io/spacekit/docs/).

IAU Minor Planet Center. n.d. "IAU Minor Planet Center." Retrieved November 14, 2021 (https://minorplanetcenter.net/data).

Jiang, Zhang. 2015. "GIXSGUI: A MATLAB Toolbox for Grazing-Incidence X-Ray Scattering Data Visualization and Reduction, and Indexing of Buried Three-Dimensional Periodic Nanostructured Films." *Journal of Applied Crystallography* 48(3):917–26. doi: 10.1107/S1600576715004434.

Johnson, Lindley. 2016. "Planetary Defense Coordination Office (PDCO)." in *URL www. lpi. usra. edu/sbag/meetings/jan2017/presentations/Johnson. pdf*.

de Juan, Anna, and R. Tauler. 2019. "Chapter 8 - Data Fusion by Multivariate Curve Resolution." Pp. 205–33 in *Data Handling in Science and Technology*. Vol. 31, *Data Fusion Methodology and Applications*, edited by M. Cocchi. Elsevier.

Jusoh, Shaidah, and Sufyan Almajali. 2020. "A Systematic Review on Fusion Techniques and Approaches Used in Applications." *IEEE Access* PP:1–1. doi: 10.1109/ACCESS.2020.2966400.

Kaasalainen, M., and J. Torppa. 2001. "Optimization Methods for Asteroid Lightcurve Inversion: I. Shape Determination." *Icarus* 153(1):24–36. doi: 10.1006/icar.2001.6673.

Kaasalainen, M., J. Torppa, and J. Piironen. 2002. "Models of Twenty Asteroids from Photometric Data." *Icarus* 159(2):369–95. doi: 10.1006/icar.2002.6907.

Kaasalainen, Mikko, Josef Ďurech, and Vojtěch Sidorin. 2014. "DAMIT: Database of Asteroid Models from Inversion Techniques." *Astrophysics Source Code Library* ascl:1412.004.

Kaasalainen, Mikko, Petr Pravec, Yurij N. Krugly, Lenka Šarounová, Johanna Torppa, Jenni Virtanen, Sanna Kaasalainen, Anders Erikson, Andreas Nathues, and Josef Ďurech. 2004. "Photometry and Models of Eight Near-Earth Asteroids." *Icarus* 167(1):178–96.

Kim, Ho-Jun, Eun-Jeong Ko, Young-Ho Jeon, and Ki-Hoon Lee. 2020. "Techniques and Guidelines for Effective Migration from RDBMS to NoSQL." *Journal of Supercomputing* 76(10):7936–50. doi: 10.1007/s11227-018-2361-2.

Kota, Rajesh, and R. Oehler. 2005. "Horus: Large-Scale Symmetric Multiprocessing for Opteron Systems." *IEEE Micro* 25(2):30–40. doi: 10.1109/MM.2005.28.

Larson, Stephen. 2006. "Current NEO Surveys." *Proceedings of the International Astronomical Union* 2(S236):323–28.

Lodha, Suresh K., and Richard Franke. 1997. "Scattered Data Techniques for Surfaces." Pp. 181–181 in *Scientific Visualization Conference (dagstuhl'97)*. IEEE.

Mainzer, Amy, J. Bauer, R. M. Cutri, T. Grav, J. Masiero, R. Beck, P. Clarkson, T. Conrow, J. Dailey, and P. Eisenhardt. 2014. "Initial Performance of the NEOWISE Reactivation Mission." *The Astrophysical Journal* 792(1):30.

Michel, Patrick, Andrew Cheng, Michael Küppers, Petr Pravec, J. Blum, Marco Delbo, S. F. Green, P. Rosenblatt, K. Tsiganis, and Jean-Baptiste Vincent. 2016. "Science Case for the Asteroid Impact Mission (AIM): A Component of the Asteroid Impact & Deflection Assessment (AIDA) Mission." *Advances in Space Research* 57(12):2529–47.

Müller, T. G., T. Sekiguchi, M. Kaasalainen, M. Abe, and S. Hasegawa. 2005. "Thermal Infrared Observations of the Hayabusa Spacecraft Target Asteroid 25143 Itokawa." *Astronomy & Astrophysics* 443(1):347–55. doi: 10.1051/0004-6361:20053862.

multiprocessing. n.d. "Multiprocessing — Process-Based Parallelism — Python 3.10.0 Documentation." Retrieved November 12, 2021 (https://docs.python.org/3/library/multiprocessing.html).

Nachouki, Gilles, and Mohamed Quafafou. 2008. "Multi-Data Source Fusion." *Information Fusion* 9(4):523–37.

Napier, W. M. 2015. "Giant Comets and Mass Extinctions of Life." *Monthly Notices of the Royal Astronomical Society* 448(1):27–36. doi: 10.1093/mnras/stu2681.

NASA Open Data Portal. n.d. "NASA Open Data Portal." *NASA*. Retrieved November 14, 2021 (https://data.nasa.gov/browse).

NICA, ELVIRA, BOGDAN GEORGE TUDORICA, DOREL-MIHAIL DUSMANESCU, GHEORGHE POPESCU, and ALINA MARIA BREAZ. 2019. “Databases Security Issues - A Short Analysis on the Emergent Security Problems Generated By NoSQL Databases.” *Economic Computation and Economic Cybernetics Studies and Research* 53(3/2019):113–29. doi: 10.24818/18423264/53.3.19.07.

Niedermayer, Johannes, Andreas Züfle, Tobias Emrich, Matthias Renz, Nikos Mamoulis, Lei Chen, and Hans-Peter Kriegel. 2013. “Probabilistic Nearest Neighbor Queries on Uncertain Moving Object Trajectories.” *Proceedings of the VLDB Endowment* 7(3):205–16. doi: 10.14778/2732232.2732239.

Ntumba, Manuel, Saurabh Gore, and Jean-Baptiste Awanyo. 2021. “Prediction of Apophis Asteroid Flyby Optimal Trajectories and Data Fusion of Earth-Apophis Mission Launch Windows Using Deep Neural Networks.” *ArXiv:2104.06249 [Astro-Ph]*. doi: 10.13140/RG.2.2.26280.49924.

NumPy. n.d. “NumPy.” Retrieved November 12, 2021 (https://numpy.org/).

Pelton, Joseph N. 2021. “Asteroids and Planetary Protection Systems.” Pp. 103–19 in *Space Systems and Sustainability*. Springer.

Perna, D., M. A. Barucci, Line Drube, A. Falke, M. Fulchignoni, A. W. Harris, and Z. Kanuchova. 2015. “A Global Response Roadmap to the Asteroid Impact Threat: The NEOShield Perspective.” *Planetary and Space Science* 118:311–17.

Perozzi, E., R. Binzel, A. Rossi, and G. B. Valsecchi. 2010. “Asteroids More Accessible than the Moon.” *EPSC* 5:750.

Perozzi, Ettore, Alessandro Rossi, and Giovanni B. Valsecchi. 2001. “Basic Targeting Strategies for Rendezvous and Flyby Missions to the Near-Earth Asteroids.” *Planetary and Space Science* 49(1):3–22.

Pope, Kevin O., Kevin H. Baines, Adriana C. Ocampo, and Boris A. Ivanov. 1994. “Impact Winter and the Cretaceous/Tertiary Extinctions: Results of a Chicxulub Asteroid Impact Model.” *Earth and Planetary Science Letters* 128(3–4):719–25.

Pope, Kevin O., Kevin H. Baines, Adriana C. Ocampo, and Boris A. Ivanov. 1997. “Energy, Volatile Production, and Climatic Effects of the Chicxulub Cretaceous/Tertiary Impact.” *Journal of Geophysical Research: Planets* 102(E9):21645–64. doi: 10.1029/97JE01743.

Popova, Olga P., Peter Jenniskens, Vacheslav Emel’yanenko, Anna Kartashova, Eugeny Biryukov, Sergey Khaibrakhmanov, Valery Shuvalov, Yurij Rybnov, Alexandr Dudorov, Victor I. Grokhovsky, Dmitry D. Badyukov, Qing-Zhu Yin, Peter S.

Gural, Jim Albers, Mikael Granvik, Läslo G. Evers, Jacob Kuiper, Vladimir Kharlamov, Andrey Solovyov, Yuri S. Rusakov, Stanislav Korotkiy, Ilya Serdyuk, Alexander V. Korochantsev, Michail Yu. Larionov, Dmitry Glazachev, Alexander E. Mayer, Galen Gisler, Sergei V. Gladkovsky, Josh Wimpenny, Matthew E. Sanborn, Akane Yamakawa, Kenneth L. Verosub, Douglas J. Rowland, Sarah Roeske, Nicholas W. Botto, Jon M. Friedrich, Michael E. Zolensky, Loan Le, Daniel Ross, Karen Ziegler, Tomoki Nakamura, Insu Ahn, Jong Ik Lee, Qin Zhou, Xian-Hua Li, Qiu-Li Li, Yu Liu, Guo-Qiang Tang, Takahiro Hiroi, Derek Sears, Ilya A. Weinstein, Alexander S. Vokhmintsev, Alexei V. Ishchenko, Phillipe Schmitt-Kopplin, Norbert Hertkorn, Keisuke Nagao, Makiko K. Haba, Mutsumi Komatsu, Takashi Mikouchi, and (THE CHELYABINSK AIRBURST CONSORTIUM). 2013. "Chelyabinsk Airburst, Damage Assessment, Meteorite Recovery, and Characterization." *Science* 342(6162):1069–73. doi: 10.1126/science.1242642.

Pravec, P., P. Scheirich, P. Kušnirák, L. Šarounová, S. Mottola, G. Hahn, P. Brown, G. Esquerdo, N. Kaiser, Z. Krzeminski, D. P. Pray, B. D. Warner, A. W. Harris, M. C. Nolan, E. S. Howell, L. A. M. Benner, J. L. Margot, A. Galád, W. Holliday, M. D. Hicks, Yu. N. Krugly, D. Tholen, R. Whiteley, F. Marchis, D. R. DeGraff, A. Grauer, S. Larson, F. P. Velichko, W. R. Cooney, R. Stephens, J. Zhu, K. Kirsch, R. Dyvig, L. Snyder, V. Reddy, S. Moore, Š. Gajdoš, J. Világi, G. Masi, D. Higgins, G. Funkhouser, B. Knight, S. Slivan, R. Behrend, M. Grenon, G. Burki, R. Roy, C. Demeautis, D. Matter, N. Waelchli, Y. Revaz, A. Klotz, M. Rieugné, P. Thierry, V. Cotrez, L. Brunetto, and G. Kober. 2006. "Photometric Survey of Binary Near-Earth Asteroids." *Icarus* 181(1):63–93. doi: 10.1016/j.icarus.2005.10.014.

Qi, Jun, Po Yang, Lee Newcombe, Xiyang Peng, Yun Yang, and Zhong Zhao. 2020. "An Overview of Data Fusion Techniques for Internet of Things Enabled Physical Activity Recognition and Measure." *Information Fusion* 55:269–80.

Quemy, Alexandre. 2019. "Data Pipeline Selection and Optimization." in *DOLAP*.

Rampino, Michael R., Ken Caldeira, and Andreas Prokoph. 2019. "What Causes Mass Extinctions? Large Asteroid/Comet Impacts, Flood-Basalt Volcanism, and Ocean Anoxia—Correlations and Cycles." *Geological Society of America Special Paper* 542:271–302.

Reddy, Vishnu, Tasha L. Dunn, Cristina A. Thomas, Nicholas A. Moskovitz, and Thomas H. Burbine. 2015. "Mineralogy and Surface Composition of Asteroids." *Asteroids IV* (2867).

Remondino, Fabio. 2003. "From Point Cloud to Surface: The Modeling and Visualization Problem." *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* 34.

Sentry. n.d. "Sentry: Earth Impact Monitoring." Retrieved November 1, 2020 (https://cneos.jpl.nasa.gov/sentry/).

Shams, Ishan, Yun Li, Jingchao Yang, Manzhu Yu, Chaowei Yang, Myra Bambacus, Ruthan Lewis, Joseph A. Nuth, Luke Oman, Ronald Leung, Bernard D. Seery, Catherine Plesko, Kevin C. Greenaugh, and Megan B. Syal. 2019. "Planetary Defense Mitigation Gateway: A One-Stop Gateway for Pertinent PD-Related Contents." *Data* 4(2):47. doi: 10.3390/data4020047.

Shang, Haibin, and Yuxin Liu. 2017. "Assessing Accessibility of Main-Belt Asteroids Based on Gaussian Process Regression." *Journal of Guidance, Control, and Dynamics* 40(5):1144–54.

Shieh, Ce-Kuen, Sheng-Wei Huang, Li-Da Sun, Ming-Fong Tsai, and Naveen Chilamkurti. 2017. "A Topology-based Scaling Mechanism for Apache Storm." *International Journal of Network Management* 27(3):e1933-n/a. doi: 10.1002/nem.1933.

Singh, Jatinder. 2011. "FigShare." *Journal of Pharmacology and Pharmacotherapeutics* 2(2):138.

Singh, Navtej, Lisa-Marie Browne, and Ray Butler. 2013. "Parallel Astronomical Data Processing with Python: Recipes for Multicore Machines." *Astronomy and Computing* 2:1–10. doi: 10.1016/j.ascom.2013.04.002.

Snyder, John S., Vernon H. Chaplin, Dan M. Goebel, Richard R. Hofer, Alejandro Lopez Ortega, Ioannis G. Mikellides, Taylor Kerl, Giovanni Lenguito, Faraz Aghazadeh, and Ian Johnson. 2020. "Electric Propulsion for the Psyche Mission: Development Activities and Status." P. 3607 in *AIAA Propulsion and Energy 2020 Forum*.

Sohan, S. M., Frank Maurer, Craig Anslow, and Martin P. Robillard. 2017. "A Study of the Effectiveness of Usage Examples in REST API Documentation." Pp. 53–61 in *2017 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*. Raleigh, NC: IEEE.

Stephan, Thomas, George J. Flynn, Scott A. Sandford, and Michael E. Zolensky. 2008. "TOF-SIMS Analysis of Cometary Particles Extracted from Stardust Aerogel." *Meteoritics & Planetary Science* 43(1–2):285–98.

Talbert, Tricia. 2017. "Double Asteroid Redirection Test (DART) Mission." *NASA*. Retrieved November 14, 2021 (http://www.nasa.gov/planetarydefense/dart).

Thusoo, Ashish, Joydeep Sen Sarma, Namit Jain, Zheng Shao, Prasad Chakka, Suresh Anthony, Hao Liu, Pete Wyckoff, and Raghotham Murthy. 2009. "Hive: A

Warehousing Solution over a Map-Reduce Framework." *Proceedings of the VLDB Endowment* 2(2):1626–29. doi: 10.14778/1687553.1687609.

Tsumoto, Shusaku, and Shoji Hirano. 2010. "Three Dimensional Trajectories Mining." Pp. 500–505 in *2010 IEEE International Conference on Systems, Man and Cybernetics*.

Usikov, Denis A. 2013. "Information and Communication Technologies (ICT) as Keys to the Enhancement of Public Awareness about Potential Earth Impacts." *Acta Astronautica* 90(1):173–79. doi: 10.1016/j.actaastro.2012.10.008.

Wagner, Sam, Alan Pitz, D. Zimmerman, and B. Wie. 2009. "Interplanetary Ballistic Missile (IPBM) System Architecture Design for Near-Earth Object Threat Mitigation." Retrieved November 8, 2020 (/paper/Interplanetary-Ballistic-Missile-(IPBM)-System-for-Wagner-Pitz/a0b6d707a1e0f9df1216d5075820ee6f03e6cc1f).

Warner, Brian D., and Robert D. Stephens. 2019. "Near-Earth Asteroid Lightcurve Analysis at the Center for Solar System Studies: 2019 January-April." *The Minor Planet Bulletin* 46(3):304.

Willis, Pascal, Nikita P. Zelensky, John Ries, Laurent Soudarin, Luca Cerri, Guilhem Moreaux, Frank G. Lemoine, Michiel Otten, Donald F. Argus, and Michael B. Heflin. 2015. "DPOD2008: A DORIS-Oriented Terrestrial Reference Frame for Precise Orbit Determination." Pp. 175–81 in *IAG 150 Years*. Springer.

Xin, Hongyi, Donghyuk Lee, Farhad Hormozdiari, Samihan Yedkar, Onur Mutlu, and Can Alkan. 2013. "Accelerating Read Mapping with FastHASH." Pp. 1–13 in *BMC genomics*. Vol. 14. Springer.

Yang, Chaowei Phil, Manzhu Yu, Mengchao Xu, Yongyao Jiang, Han Qin, Yun Li, Myra Bambacus, Ronald Y. Leung, Brent W. Barbee, Joseph A. Nuth, Bernard Seery, Nicolas Bertini, David S. P. Dearborn, Mike Piccione, Rob Culbertson, and Catherine Plesko. 2017. "An Architecture for Mitigating near Earth Object's Impact to the Earth." Pp. 1–13 in *2017 IEEE Aerospace Conference*. Big Sky, MT, USA: IEEE.

Yang, Zhouwang, Jiansong Deng, and Falai Chen. 2005. "Fitting Unorganized Point Clouds with Active Implicit B-Spline Curves." *The Visual Computer* 21(8):831–39.

Yeomans, D. K., S. R. Chesley, and P. W. Chodas. 2010. "NASA's Near-Earth Object Program Office." *ББК 22.655 340* 244.

Yeomans, Donald K., Ronald C. Baalke, A. B. Chamberlain, Stephen R. Chesley, Paul W. Chodas, and Jon D. Giorgini. 2001. "The Near-Earth Object Program Office at the NASA/Jet Propulsion Laboratory."

Yongyao Jiang. 03:59:51 UTC."A Knowledge Discovery Framework for Planetary Defense."

Zhao, Hong-Kai, Stanley Osher, Barry Merriman, and Myungjoo Kang. 2000. "Implicit and Nonparametric Shape Reconstruction from Unorganized Data Using a Variational Level Set Method." *Computer Vision and Image Understanding* 80(3):295–314.

## BIOGRAPHY

Ishan Shams is a software architect with over 11 years of experience in developing enterprise-level applications. He has been involved in the IT industry for almost half his age and shares his knowledge through books, blogs, conferences, and training.

He received his Bachelor of Science degree from George Mason University (Fairfax, Virginia) in 2016. He completed his bachelor's degree and then enrolled in a Doctorate program at the same institution. His undergrad major was focused on applied information technology. He possesses solid expertise in interactive data visualization, artificial intelligence/machine learning (AI/ML), metaverse (AR/VR), cloud computing, and no-code platforms.

He currently works as a senior solutions architect with clients across North America on various topics such as architecture design patterns for digital transformation; enterprise solution architectures; enterprise integration patterns; cloud computing; cognitive/AI-based IoT solutions, and DevOps.

As of 2022, He is currently authoring a book on the topic of Enterprise Architecture Patterns Guidebook - A Syllabus to Build Your Future, where he talks about designing future networks through architectural patterns for automation of operations tasks.

He has also managed to earn over six highly technical certifications from Microsoft. Certifications such as AZ-104: Azure Administrator, PL-100: Power Platform App Maker, PL-900: Power Platform Fundamentals, PL-200: Power Platform Functional Consultant, PL-400: Power Platform Developer, MB-200: Dynamics 365/Power Platform Functional Consultant, MB-400: Dynamics 365/Power Platform Developer, and MB-600: Dynamics 365/Power Platform Solutions Architect.