

Guided Inverse Gravity Modeling for Asteroids using Neural Networks

Jan-Frederik Stock*

Hermann Meißenhelger†
Universität Bremen

Thomas Hudcovic‡
Universität Bremen

Gabriel Zachmann§
Universität Bremen

ABSTRACT

We present a new dataset for use in training inverse gravity modeling algorithms, developed with approaches based on convolutional neural networks in mind. This dataset comprises hundreds of asteroids with homogeneous or random mass distributions and the corresponding gravitational fields. We also modify the GeodesyNet, a neural network based inverse gravity modeling technique, to be able to incorporate additional information into the gravity inversion process, apart from the gravitational field. We call this “guiding” the GeodesyNet, and compare it to a more classical inversion method, for which we also implement guidance. Both methods are able to generate mass distributions that match the input gravitational field well, while incorporating additional guidance information.

Index Terms: Neural Networks, Inverse Gravity Modeling, GeodesyNet, Asteroids, Gravitational Fields, Mass Distribution.

1 INTRODUCTION

In the recent decades, there have been multiple space missions with the goal to visit and gather data about small-bodies in our solar system. These include the NEAR mission to the asteroid 433 Eros [11], Hayabusa to the asteroid Itokawa [4], the Dawn mission to the proto-planets Vesta and Ceres [7] and most recently the OSIRIS-REx mission to the asteroid Bennu. Some of these missions were able to get close enough to the body to measure detailed gravitational potentials. The gravitational potential outside of a body is produced by its internal mass distribution, which is why it can be used to reconstruct the internal mass distribution of a body. This process is called gravity inversion [1]. Studying the internal structure of a small-body can give insight into its history and how the body has changed over time. Knowledge about the internal structure also helps to build more accurate models of small bodies [20]. Information about the internal mass distribution of a body can also be obtained through other methods than analyzing the gravitational field, for example through radar measurements or analysis of collected dust particles [17]. Recently, a gravity inversion method based on Neural Networks has been developed, which requires fewer assumptions about the internal mass distribution and shape of a body, and can yield more accurate results in some situations, compared to classical inversion methods [6]. There also have been recent works that employ Machine Learning based approaches to perform gravity inversion on earth [23] [9] [24]. Our goal in this work is to incorporate available information about the mass distribution of a small-body into the gravity inversion process of the Machine Learning based method most suited to this task, and thus obtain a mass distribution that is consistent with this information.

*e-mail: jastock@uni-bremen.de

†e-mail: hmeiss51@informatik.uni-bremen.de

‡e-mail: hudo@uni-bremen.de

§e-mail: zach@cs.uni-bremen.de

2 FORWARD MODELING OF GRAVITATIONAL FIELDS

In order to do inverse gravity modeling, gravitational data which the chosen inverse gravity modeling method should try to match as closely as possible, is needed. In the case of a space mission like NEAR visiting the asteroid Eros [19] or more recently OSIRIS-REx visiting the asteroid Bennu [8], this data might come from a spacecraft taking measurements around the body of interest. The other option is to use data generated through forward modeling, where the gravitational field around a body is calculated from the shape of the body as well as an assumption about the distribution of mass inside the body. For the development of inverse gravity modeling methods this approach is suited well, because it allows the easy generation of multiple variants of mass distributions and their corresponding gravitational fields to test with. Data generated by this method is also much more easily accessible as gravitational data measured by spacecraft.

Commonly used methods to calculate gravitational fields through forward modeling include using Spherical Harmonics, the Polyhedral Method and the Mass Concentration Method. Spherical Harmonics have issues for bodies which differ strongly from a sphere in shape, like asteroids often do. For details see the work of Heiskanen and Moritz [5], Werner [21] and Sebera et al. [15]. The polyhedral method does not suffer from this issue and provides an exact solution at any point outside of the body [21], but is computationally expensive when adapted to heterogeneous bodies [18]. This property makes it unsuitable for this work, since the main objective is to study heterogeneous bodies. The mascon method can get close to the solution provided by the polyhedral method [12] while making it easy to model heterogeneous bodies, which is why we chose it for this work.

When calculating the gravitational field of a body using the mass concentration method, the body is approximated as a set of point masses. The gravitational acceleration or potential at a given point $\mathbf{p}$ is then derived by summing up the contributions of every point mass.

$$\mathbf{a} = -G \sum_i \frac{\rho_i V_i}{r_i^3} \mathbf{r}_i \quad (1)$$

Equation 1 shows the formula for the gravitational acceleration, with G denoting the gravitational constant, ρ_i and V_i the density and volume of the mass concentration point or mascon i and with r_i denoting the distance between the mascon $\mathbf{x}_i$ and the sample point $\mathbf{p}$ [13]. The vector $\mathbf{r}_i$ runs from the sample point $\mathbf{p}$ to the mascon $\mathbf{x}_i$ and can be calculated as shown in equation 2 [14].

$$\mathbf{r}_i = \mathbf{x}_i - \mathbf{p} \quad (2)$$

Implementations of the mascon method differ in how the body of interest is subdivided into smaller sections and how these sections are placed throughout the body. Park et al. [12] work with uniform distribution, while Meißenhelger et al. [10] utilize sphere packing to achieve smaller errors near the bodies surface. Chanut et al. [2] propose a solution where they subdivide a polyhedral shape model to determine mascon locations and masses.

3 RELATED WORK

There are two kinds of approaches used to do inverse gravity modeling with machine learning in previous work. The *Convolutional*

Neural Network based approach, exemplified by the works of Wu et al. [23], Li et al. [9] and Yu-Feng et al. [24], and the *NeRF inspired approach* presented by Izzo and Gómez [6].

3.1 Inverse Gravity Modeling with Convolutional Networks

Approaches which employ convolutional networks to do inverse gravity modeling usually apply their method to gravitational measurements on the surface of the earth, to predict the density in the earths crust underneath. An example for this is the work of Wu et al. [23], where the authors train a U-Net and apply it to gravity data from East Antarctica as well as synthetic data. The model is able to reconstruct the general shape of high densities regions in the synthetic data well, although the density values are not exact, especially when more complex shapes are tested. The model is more exact in regions closer to the surface, with higher divergences in the deeper sections. When applying their method to real data from East Antarctica, the authors find that their model yields a density distribution which is consistent with previous studies [23].

Li et al. [9] use a similar method, the main difference being the addition of residual connections to a U-Net based approach. They attain slightly better results in comparison to a U-Net based approach. Another similar approach is presented by Yu-Feng et al. [24], where the authors use a similar data creation method, but use a 3D-UNet instead of the two dimensional version. Results are difficult to compare with the other presented works, as F1-Scores are given instead of attained errors.

The models developed in these works perform well on the applications on earth for which they were designed. In the context of this work however, they would have to be adapted to work on small-bodies in space. The main difference between these applications is the kind of gravitational data available. Where in the case of earth the data is usually measured or generated in a two dimensional plane with the mass distributed underneath, measurements in space are usually taken by space probes in orbit around a body of interest, see for example Lauretta et al. [8]. This leads to measurements in the shape of spherical trajectories around the body, and this spatial information should be utilized when adapting approaches based on convolutional networks. If input data for a network to predict a small-bodies density distribution were limited to a two dimensional plane on one side of the body, the model would probably yield less accurate results in the parts farthest away from the gravitational measurements, as could be seen in Wu et al. [23]. There have been approaches to adapt the U-Net to work on spherical data, for example in Zhao et al.[25], where the authors adapt the convolution operation to spherical surfaces and build a Spherical U-Net with it. The concept of the 3D U-Net as proposed by Çiçek et al. [26] and adapted to gravitational inversion in the work of Yu-Feng et al. [24] is also interesting in the context of this work, as the output of the inversion is a density distribution in three dimensional space. The inversion might thus benefit from the additional 3D processing capabilities the 3D U-Net provides. Ideally, a U-Net based network for gravity inversion on small-bodies would process the spherical gravitational data of the input with spherical convolutions in the encoder part of the network, similar to [25]. In the decoder 3D operations similar to the ones in [26] would be used to arrive at the three dimensional output of the density distribution. The problem with this approach is the fact that the architecture of the U-Net is symmetrical, which results in the shape of the inputs and outputs being usually similar. For example, the Spherical U-Net takes a spherical surface as an input and also yields a spherical surface as an output [25]. Similarly the 3D U-Net gets three dimensional input data and outputs three dimensional data. Wu et al. are able to generate three dimensional output from two dimensional input by encoding the third dimension in the channels of their output image, but the data flowing through the U-Net is still two-dimensional

[23]. This is important because one of the features of the U-Net is the concatenation of features from the encoding part with features from the decoding path. This would be difficult to accomplish with two representations in the encoder and decoder part of the network that are as different as the spherical representation of the Spherical U-Net and the three dimensional representation of the 3D U-Net. Due to these challenges we chose to focus on the NeRF inspired approaches instead, which will be presented in the following.

3.2 Inverse Gravity Modeling with a NeRF inspired approach

Izzo and Gómez propose a solution to the inverse gravity modeling problem, where they train a neural network to represent a mapping from Cartesian coordinates to the body density [6], a representation similar to the one used by NeRFs. The input in the form of Cartesian coordinates is fed into the network, flowing through a number of fully connected layers with the output of the network being the predicted density at the input point. An arbitrary number of points and their corresponding densities can be sampled from the network like this. The gravitational field corresponding to these densities can then be calculated and compared to either a gravitational field obtained through forward modeling or actual measurements of a gravitational field. The error between the predicted and ground-truth gravitational field is calculated and used in backpropagation to update the neural networks parameters, to minimize the error. This makes the gravitational field the predicted density produces more closely resemble the ground-truth gravitational field step by step. Notably, the GeodesyNet does not require a shape model to be given and can learn the shape of the body from the gravitational data. However, GeodesyNets can incorporate a shape model for improved results.

Izzo and Gómez compare their method to the work of Wittick and Russell [22], where a hybrid mascon and spherical harmonics method is used. In this work, the parameters of the model are determined using least-squares estimation. The GeodesyNet is found to reach comparable accuracies to some of those presented by Wittick and Russell [22], even though no shape model is used. Izzo and Gómez note that the usage of a shape model with the GeodesyNet “would ... result in orders of magnitude smaller errors”[6, Supplementary Method 5]. They point out that it is generally difficult to compare results of gravity modeling across the literature, as a common validation practice does not exist. To allow for better quantitative comparison, the authors implement their own mascon approach, which does not require shape information. When looking at the performance farther away, both models perform well, with the better performing method depending on the studied body. Closer to the surface, the GeodesyNet performs consistently better than the mascon method. The authors also test the performance of the GeodesyNet when utilizing a shape model on the heterogeneous ground-truth data. They find that this improves the error in comparison to not utilizing a shape model, especially at low altitudes above the bodies surface. Utilizing the shape model also results in a density distribution that is closer to the ground-truth density distribution.

4 DATASET

In anticipation of utilizing an approach based on a convolutional neural networks to do inverse gravity modeling as described in section 3.1, we created a training dataset for such an approach. The dataset consists of a number of asteroid density distributions and their corresponding gravitational fields. It is based on the shape models and additional information about small-bodies provided on the 3D Asteroid Catalogue website [3]. The shape models and information on this website were gathered from different sources, for more details refer to the sources provided on the pages for the different small bodies on the website.

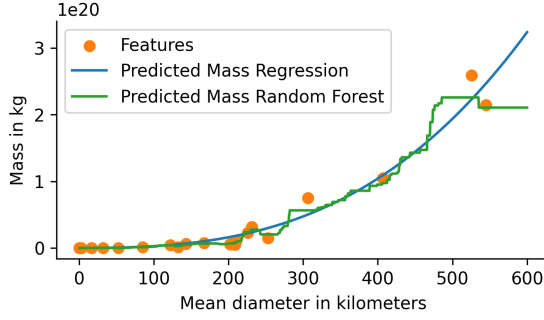


Figure 1: Predicted body mass with Polynomial Regression and Random Forest using the mean diameter.

We noticed that only 26 out of the 1635 asteroids came with information about their mass, although 1563 came with a diameter. Since the total mass of a body is needed to calculate a gravitational field that is scaled correctly, we decided to reconstruct the masses of the bodies by using the diameter. We assumed that the volume and total mass of a body would be roughly correlated. Since the volume and radius of a sphere are in a cubical relation we fit a polynomial of degree three to the available data. To make sure that the resulting function would yield a mass of zero for a mean diameter of zero, we disabled the fitting of the intercept. We also forced the regression to have positive coefficients to make sure positive diameters would always yield positive masses. The result of this experiment is shown in figure 1. The regression fits the data well and seems to represent the relation between mean diameter and body mass meaningfully. We also tested other regression methods like the Random Forest. We were able to get a better score with the Random Forest in comparison to the Polynomial Regression when we split the available data with mass into a train and test set, but in a case with so little training data the Random Forest is prone to overfit. It looks like this happened when looking at the plot of the Random Forests predictions in figure 1. To avoid overfitting, in this case the simpler model which can be more easily derived from the real physical relationship of these two properties is the better choice.

4.1 Modeling and calculating the gravitational field

With the masses for most of the asteroids restored, we started working on the mascon method to calculate the asteroids gravitational fields. After loading an asteroids mesh, we scale the maximum extent of the asteroid to be its diameter, since the real extents in x, y and z direction are again only available for a limited number of asteroids. This is only an approximation of the actual size of the asteroid, but the size is in the correct order of magnitude. The next step is the subdivision of the asteroid into multiple mascons. As we were building the dataset in anticipation of training a model like the ones presented in section 3.1, our aim was to create mascon models with the same dimensions for all asteroids. These methods require all of their inputs to have the same dimensions. Mascon models usually produce different numbers and volumes of mascons, depending on the shape of the modeled body. Our initial solution to this problem was to fill the bounding box of the asteroid with mascons shaped like cuboids. The length of each cuboid in each dimension is the length of the bounding box in this dimension divided by the desired number of subdivisions per dimension. This leads to the volume of the bounding box being completely covered by mascons. Figure 2 shows this method with ten subdivisions leading to 1000 mascons. The advantage of this method is the relatively small number of mascons which do not overlap with the asteroid while always providing the same dimensions no matter the shape of the

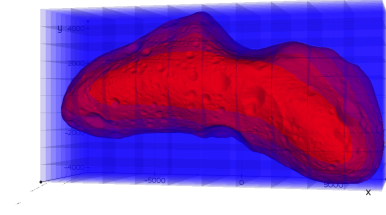


Figure 2: Filling the bounding box with mascons. Here ten subdivisions were used, leading to 1000 mascons.

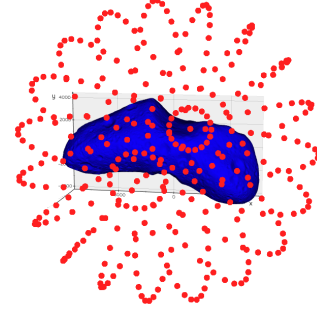


Figure 3: The sampled points creating a sampling sphere around the asteroid.

asteroid. A potential issue is the fact that information about the position and proportions of the mascons gets lost when the mascons masses are the only input to a machine learning model. This leads to the property that an asteroid can be scaled in different dimensions and will still yield the same mascon representation from the view of a machine learning model, as long as the volume of the mascons stays the same. For example, an asteroid with a bounding box of dimensions (10, 1, 1) will yield the same representation as the same asteroid rescaled to a bounding box of (1, 10, 1). To address this issue, we implemented a different subdivision method, which is similar in layout to the mascon based approach by Izzo and Gómez [6], which they compare the GeodesyNet to. With this approach, we place the asteroid into a cube, which is then subdivided into a grid of cubes. This method leads to the same number of mascons for all asteroids as well, but the mascons are all cubes and therefore have the same relative dimensions across asteroids. This method also does not have the scaling invariance problem discussed before. A potential drawback of this method might be the greater number of mascons that do not overlap with the asteroid. This leads to a lower resolution of mascons covering the asteroid compared to subdividing the bounding box with the same total number of mascons. Both of these methods need to be compared in practice, but subdividing the cube has fewer theoretical issues.

After subdividing, densities can be assigned to all of the mascons. We implemented both a uniform distribution of density, as well as a random one. For the uniform distribution, the total mass is divided equally between the mascons whose centerpoints lie inside of the asteroid. We then calculate the density by dividing this mass by the volume of each mascon. For the random distribution we sample a random value from the continuous uniform distribution in the interval [0, 1) for every mascon whose centerpoint lies within the asteroid. We then scale these values so their sum equals the total mass of the asteroid. The densities are then calculated again by dividing the masses by the mascons volume.

To be able to calculate the gravitational field corresponding to the density distributions, we create a sphere of points at which the gravitational acceleration or potential can be calculated. To describe the points on the sphere we use spherical coordinates similar to the ones

described bei Heiskanen and Moritz [5, p. 18]. A point P on the sphere is described by its radius r from the center of the sphere, as well as the angles λ and θ . The angle λ describes the angle between P and the x-axis in the x-y plane and the angle θ is the angle between the line connecting the center of the sphere and the point P and the z-axis. By subsampling the angles λ and θ we create evenly spaced rings of points around the z-axis, which are themselves evenly spaced in the range of θ . Figure 3 shows the resulting sampling sphere. With the mascons and the sampling points determined, we calculate the gravitational accelerations or potentials with equation 1 or the corresponding formula for potentials.

4.2 Usage

After creating the dataset we decided to implement the incorporation of additional information into the inversion process, based on the GeodesyNet by Izzo and Gómez [6]. Since this method does not need the amount of data in this dataset to be trained, we were unfortunately unable to use this dataset in this work. It can still serve as a basis for future work, for example to train gravity inversion models like the ones described in section 3.1.

5 MODIFYING THE GEODESYNET

The aim of this work is to achieve an inverse gravity modeling method whose performance is comparable to other inversion methods and which is able to incorporate additional expert information into the inversion process, in the form of regions inside of the asteroid with predefined density. We have presented different machine learning based methods for inverse gravity modeling in sections 3.1 and 3.2 and decided to use the most promising for asteroids, which are the GeodesyNets introduced by Izzo and Gómez [6]. In addition to this approach already working for asteroids and the shape of gravitational information commonly used with them, we also chose this approach because it seemed the most straightforward to implement the introduction of additional information into the inversion process for. We will refer to methods which incorporate regions of predefined density as guided inverse gravity modeling from hereon.

5.1 The Loss Function

In the unmodified form implemented by Izzo and Gómez [6], a training iteration of the GeodesyNet starts with sampling its output on a grid inside the hypercube with dimensions $[-1, 1]^3$. The gravitational acceleration or potential is then calculated at a number of target points outside of the asteroid using the densities sampled from the GeodesyNet. This gravitational field represented by the GeodesyNet is used to calculate the loss of the neural network together with the gravitational field calculated from the mascon ground-truth. After comparing a number of different loss functions, Izzo and Gómez [6] settle on a modified version of the Mean Absolute Error. They introduce a mass normalization factor κ which is multiplied with the networks predictions to scale them. This allows the network to focus on learning the difference to a homogeneously filled volume and not on finding the correct absolute mass of the body as well. The value of κ is calculated analytically in each training iteration of the network and the authors find the optimal value for it to be

$$\kappa = \frac{\sum_{i=1}^n \hat{y}_i y_i}{\sum_{i=1}^n y_i^2} \quad (3)$$

with $y_i, i = 1..n$ being the ground-truth accelerations or potentials from the mascon model and $\hat{y}_i, i = 1..n$ the predicted accelerations or potentials of the network [6, Supplementary Method 1]. When incorporated into the Mean Absolute Error, the resulting loss function is

$$L_{\kappa MAE} = \frac{1}{n} \sum_{i=1}^n |y_i - \kappa \hat{y}_i| \quad (4)$$

To implement the guided inverse gravity modeling for the GeodesyNets, we took inspiration from an optional part of the training implemented by Izzo and Gómez called Vision Loss. The Vision Loss is an additional term that is added to the loss calculated on the gravitational field. It is calculated by sampling a number of points outside of the body, for which a shape model is needed. The expectation is for these points to have a mass of zero, as they are outside of the body. Deriving the Vision Loss now involves calculating the Mean Absolute Error between the sampled points and zero. The result is added to the loss calculated on the gravitational field to form the total loss. This urges the densities at the points to be zero.

To implement guided inverse gravity modeling, we added a similar term to the loss based on the gravitational field, which we call the guidance loss. The predefined regions of density can be given by the user either in the form of spheres or a plane that splits the asteroid into two halves. The spheres are characterized by their centerpoints and radii, as well as densities which define the desired densities in the volumes covered by the spheres. The plane is defined by a point in the plane and the planes normal vector. The density corresponding to the plane describes the desired density on the positive side of the plane. We calculate the guidance loss by first sampling a number of points inside of the spheres or inside the asteroid on the positive side of the plane. The output of the GeodesyNet is sampled at these points and the guidance loss calculated using

$$L_{\kappa G} = \frac{1}{n} \sum_{i=1}^n |y_{\delta G} - \kappa \hat{y}_i| \quad (5)$$

with $y_{\delta G}$ representing the predefined guidance density and $\hat{y}_i, i = 1..n$ the predicted densities at a number of points inside of the predefined regions or region. The mass normalization factor κ is also used to scale the networks predictions, so that the densities $\hat{y}_i$ are correct. Incorporating the guidance loss into the total loss leads to the following equation for the total loss:

$$L = L_{\kappa MAE} + L_{\kappa G} \quad (6)$$

Initial testing with this approach revealed that it performed worse when compared to the ground-truth based on the mascon model than the unmodified GeodesyNet, which does not incorporate additional density information. We looked at how the two components of the loss, gravitation based and guidance density based, developed over the duration of a training run. Figure 4 shows both loss components as well as the loss of an unmodified GeodesyNet for the last 2000 iterations of the run. It looked to us like the network was optimizing the guidance loss at the detriment of the gravitational field based loss, which would explain the increased error with regard to the ground-truth data. To test this hypothesis and correct this potential issue, we introduced a factor $g, 0 \leq g \leq 1$ to be able to scale both losses. This results in the modified equation

$$L = (1 - g) \cdot L_{\kappa MAE} + g \cdot L_{\kappa G} \quad (7)$$

to calculate the total loss. Introducing the factor and finding a suitable value for it through empirical means lead to all three losses being much closer in magnitude. The loss based on the gravitational field is now an order of magnitude smaller and is close to the loss of the unmodified GeodesyNet. The guidance loss nearly doubles after introducing the scaling factor when comparing the iterations with the smallest overall loss across both runs. This is a worthwhile trade-off for the improvement in the gravitational field based loss.

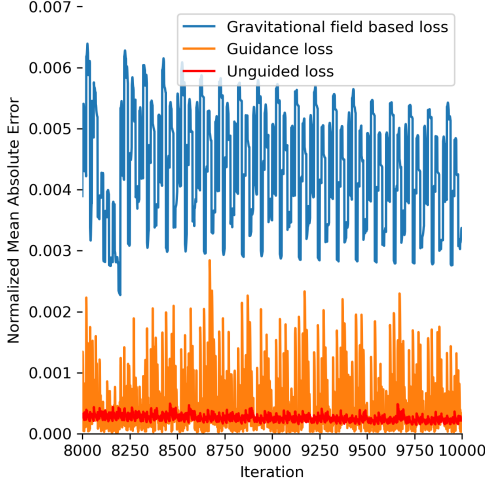


Figure 4: Comparing the last iterations of a training run with the initial implementation. The separated components of the loss are shown, one based on the ground-truth gravitational field and the other on the given guidance density for one guidance volume, with the loss of an unmodified GeodesyNet depicted for comparison.

With the addition of scaling the losses the guidance loss works well and achieves the goal of being able to incorporate expert information and previous knowledge about the density in certain regions into the inversion process of the GeodesyNet.

The optimal scaling factor varies between different bodies and guidance regions. Since it is impractical to do an empirical study for every new body and guidance region to find the optimal scaling factor, we introduced a method to scale the guidance loss automatically. As described before, introducing a good scaling factor found through empirical means leads to the gravitational field based loss and the guidance loss ending in the same order of magnitude. To force this throughout the whole training run, we scale the guidance loss to always be the same absolute number as the gravitational field based loss:

$$L_{SG} = \left(\frac{L_{\kappa MAE}}{L_{\kappa G}} \right) \cdot L_{\kappa G} \quad (8)$$

The resulting automatically scaled guidance loss L_{SG} is then added to the gravitational field based loss to form the total loss.

$$L = L_{\kappa MAE} + L_{SG} \quad (9)$$

5.2 Heterogeneous Ground Truth Data

To accurately reflect the use-case that a scientist with expert knowledge might use the guidance density to specify areas that in fact match the actual density of an asteroid, we created versions of the ground-truth data that reflect the predefined guidance densities. Izzo and Gómez provide mascon models with homogeneous mass distributions for “the asteroids 433 Eros, 25,143 Itokawa, and 10,1955 Bennu and the comet 67P Churyumov–Gerasimenko, as well as a fictitious Planetesimal and a toroidal-shaped body (...) [they] call Torus” [6]. For Bennu, Itokawa and Planetesimal the authors provide models with heterogeneous mass distributions as well. We chose to work on the model for the comet 67P Churyumov-Gerasimenko, as well as the asteroid Bennu. Churyumov-Gerasimenko has a density of about 1.7, Bennu of 0.7, in the unmodified version. Our first heterogeneity is a small sphere in the middle of Churyumov-Gerasimenko, with a density of 1, for

Bennu we placed a small sphere of density 1.2 close to the surface. The second type of heterogeneity was created by placing three spheres of different sizes inside of the asteroid. For Churyumov-Gerasimenko we generated two versions of this, one with densities 1, 0.5 and 2.5 for the spheres, the other with density 0 for all spheres, to simulate cavities inside the body. For Bennu we choose densities of 1, 0.5 and 1.5. To test specifying the guidance density by a plane and evaluate a heterogeneity much bigger proportionally to the rest of the asteroid, we introduced a heterogeneity for the head of the comet similar to the way Izzo and Gómez did for the asteroid Itokawa. The result can be seen in figure 5. We generated a similar version for Bennu, with the guidance density being specified for about a third of the asteroids volume.

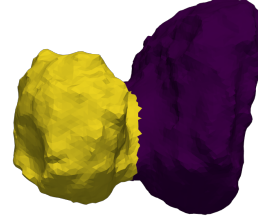


Figure 5: 67P Churyumov-Gerasimenko with a density heterogeneity introduced for its head.

The ground-truth is generated by subdividing the interior volume of the surface mesh of a given small body into tetrahedrons, using TetGen [16]. A mascon is placed at the center of each tetrahedron and a mass proportional to the tetrahedrons volume is assigned to it, with all masses totalling the bodies mass, in case of a homogeneous mass distribution. In the version used by Izzo and Gómez, this method results in a mascon model with about 57000 mascons for the comet 67P Churyumov-Gerasimenko, and about 38000 for Bennu. To create the heterogeneous versions of the bodies, we first selected all tetrahedrons whose centers overlapped with any of the guidance spheres or are on the positive side of the plane defining the guidance area. We then assigned a mass to each mascon, so that the density calculated with respect to the tetrahedrons volume matched the guidance density. This process results in either a loss or a gain of total mass of the body, as mass is removed or added to the mascons in question. To bring the total bodies mass back to its state before the modifications, we removed or added a proportionally equal amount of mass to each mascon that was not affected by the modifications. While modifying the generation of the mascon model to create the heterogeneity of the small sphere in the middle of the body for Churyumov-Gerasimenko, we noticed that the volume of the tetrahedrons whose centerpoints overlap with the sphere is about double the volume of the sphere, leading to a mass heterogeneity about double the size it should be. The reason for this is the relatively coarse resolution of tetrahedrons on the inside of the body compared to the spheres size, leading to only two tetrahedrons overlapping with the sphere which are unable to accurately approximate the sphere. To solve this problem we increased the number of tetrahedrons used to subdivide the body, leading to a model with about one million mascons for Churyumov-Gerasimenko and about 175000 for Bennu. The higher resolution leads to more tetrahedrons centerpoints overlapping with the sphere which results in a better approximation and a volume of the tetrahedrons that is much closer to the volume of the sphere.

5.3 Performance Considerations

The results in this work were mostly generated with 10000 iterations of training with a batch size of 1000 points (samples of the gravitational field around the asteroid) and 300000 integration

points sampling the density inside the unit cube, to allow for comparisons with the results of Izzo and Gómez [6]. Testing these parameters with the unmodified code provided by the authors, a full training run took eight to nine hours. The machine we were testing with was equipped with a Nvidia RTX 2080 Super, an AMD Ryzen 3900X and 32GB of RAM. In one of the notebooks from the code accompanying [6] the authors write that a full training should take roughly one hour on a Nvidia RTX 2080ti. While investigating the codebase for bottlenecks, we found that the most computationally intensive part of a training iteration by far is the calculation of the gravitational field at the target points around the asteroid based on the densities sampled in the unit cube. We adapted this code so that the gravitational accelerations are calculated in batches, allowing for multiple target points to be processed at the same time on the GPU. With enough GPU memory to process all target points in one batch, this leads to a speedup greater than 2x, reducing the time for a full training run down to between three and a half and four hours, tested on a Nvidia V100 GPU with 32GB of GPU memory. On GPUs with less memory like the 8GB of the RTX 2080 super, the batch size needs to be carefully selected, so that the data fits into the GPUs memory, while still allowing the maximum number of points to be processed at the same time. We have found processing the points in 25 batches, leading to a batch size of 40 points to work well on the RTX 2080 super. Interestingly, processing the points in batches also leads to a speedup on GPUs where the full data would fit into the memory at the same time. Using a batch size of 40 points leads to a final training time of between two and three and a half hours, depending on the guidance regions used.

5.4 Post Training

The training time of a GeodesyNet of three and a half to four hours makes it difficult for a user to test different guidance regions quickly and see which one fits the given gravitational field best. To address this issue we split the training of the GeodesyNet into two steps. First an unmodified GeodesyNet gets trained for 10000 iterations to fit the given gravitational field, without specifying any guidance regions. The resulting model serves as the base for the next step, where the training is continued with the addition of guidance regions. To evaluate the performance we compare the training in two steps to a training run that is trained with the guidance loss from the start, for 10000 iterations.

5.5 Mascon based comparative method

Izzo and Gómez [6] use a mascon based approach they call masconCUBE, to evaluate the performance of the GeodesyNet in comparison to, as discussed in section 3.2. We adapted this approach to be a suitable comparative method for the GeodesyNet with guidance, by implementing guidance for this approach as well. We implemented a guidance loss similar to the one for the GeodesyNet, described in section 5.1. To calculate the loss we first select all points of the masconCUBE which lie inside the guidance regions. We then calculate the corresponding densities for the mascon masses at these points by dividing by the volume of a mascon V_{mascon} . With the densities, we can calculate the guidance loss using the Mean Absolute Error:

$$L_{\kappa G} = \frac{1}{n} \sum_{i=1}^n \left| \kappa \cdot \frac{m_i^2}{V_{mascon}} - \rho_G \right| \quad (10)$$

The mascon masses are represented by $m_i, i = 1..n$, and the guidance density by ρ_G . The scaling factor κ is used like in the GeodesyNet to scale the predictions to their correct absolute value. The guidance losses for all guidance regions are added up and then added to the gravitational field based loss, using a guidance factor determined from the results of 6.2. Training a masconCUBE for 10000 iterations takes about an hour on a Nvidia V100 GPU.

5.6 Validation procedure

To assess the performance of the GeodesyNet and the masconCUBE and to compare the results, we use the validation procedure used by Izzo and Gómez [6]. It consists of two different approaches to selecting the validation points at which a model is compared to the ground-truth. The first approach was introduced by the authors to enable a comparison to the work by Wittick and Russell [22]. The validation points are sampled randomly at a low altitude between the surface and about 0.15 length units above the surface, as well as at a higher altitude between about 0.15 and about 0.3 length units above the surface. While testing this method on the comet 67P Churyumov–Gerasimenko, we noticed that some of the validation points ended up inside of the body, which was distorting the results. We modified the method to discard those points to remedy this problem. The other method involves sampling validation points at three different altitudes, for which we chose to use the standard values in the code provided by Izzo and Gómez. This results in 10000 points being sampled at 0.05, 0.1 and 0.25 length units above the body respectively.

We used the same number of integration points sampled in the unit cube for the forward modeling as in the training of the GeodesyNet, because the authors state in one of the provided notebooks that using a different number leads to worse results. The authors use the Normalized L1 Loss, as well as the Normalized Relative Component Loss to compare their results, which can be seen in one of the provided notebooks as well. We used the Normalized Relative Component Loss as our performance metric to compare validation results.

6 RESULTS AND DISCUSSION

In this section we will present how well the GeodesyNet and the comparable masconCUBE perform while learning a density and mass distribution for different kinds of heterogeneities, both with and without guidance. The approaches will be studied on the comet 67-P Churyumov-Gerasimenko and the asteroid Bennu. All of the results were generated while training for 10000 iterations for both the GeodesyNet and the masconCUBE, unless otherwise specified. For the GeodesyNet we used 300000 sampling points inside the unit cube for the forward modeling, and the gravitational field was evaluated at 1000 target points in every training iteration.

6.1 GeodesyNet vs. masconCUBE

For the GeodesyNets with guidance, we chose the guidance factor which produced the best performance regarding the gravitational field from section 6.2. We chose the same guidance factor for the corresponding masconCUBE model with guidance, as we expect the tradeoff between gravitational field loss and guidance loss to be similar in the masconCUBE. For the version of Churyumov-Gerasimenko with three spheres of different densities, we did not perform a full guidance factor study. Here, we used the same guidance factor that performed the best in the same three sphere version, but with the densities set to zero. In the figures, the GeodesyNets are labeled “GN” and the masconCUBEs “mC”.

When looking at the results, it becomes clear that the masconCUBE is able to match the ground-truth gravitational field far better than the GeodesyNet for the heterogeneities defined through single and multiple spheres, on both Churyumov-Gerasimenko and Bennu. An example for this can be seen in figure 6. The better performance of the masconCUBE in this regard holds true for both the guided and unguided version. On the versions of the bodies with the heterogeneities defined through a plane, the unguided masconCUBE is still the best method by far, but the errors of the masconCUBE with guidance are closer to the GeodesyNets. Here, the GeodesyNets with guidance outperform the guided masconCUBE at the low altitude intervals for both bodies. We visualized these results in figure 7.

interval, the post training model reaches the loss of the normally trained GeodesyNet and improves from thereon. For the low altitude interval, the post training has lower loss after the first 1000 iterations. For Churyumov-Gerasimenko the loss of the post-trained models gradually declines towards the loss of the normally trained guidance model, but never reaches it, for the interval altitudes and altitude 0.05. For altitude 0.1 the loss of the normally trained model gets reached after 10000 iterations of post-training, and for the altitude 0.25 this is the case after 7000 iterations.

These results suggest that this method of post-training is a worthwhile avenue for further research, to be able to get good results with guided GeodesyNets after fewer iterations of training, especially for larger guidance regions. It might be worthwhile to explore loss weighting that is specifically designed for this kind of post-training, to be able to preserve the optimization to the given gravitational field from the pre-trained model, while optimizing the guidance loss quickly.

7 FUTURE WORK

Firstly, a possible avenue for future research would be to utilize the dataset we built in section 4 to train a machine learning model similar to the ones we presented in section 3.1. It would be interesting to see how such a model would compare performance-wise to the GeodesyNets presented here, and if it would be able to generalize to unseen small bodies. If it could, this would improve upon the long training times for the GeodesyNet and masconCUBE. As an addition to this, implementing guidance for this kind of model would be interesting as well.

Building on the results of our work on the GeodesyNets, a next step would be to utilize more sophisticated techniques from multi-task learning to weight the guidance loss and the gravitational field based loss. These might be able to incorporate the guidance loss into the training, while preserving the gravitational field based loss better than the guidance factors in this work. They could also remove the necessity to choose a good guidance factor. Exploring the approach from section 5.4 further would also be worthwhile, and it might benefit from the utilization of techniques from multi-task learning as well.

8 CONCLUSION

In conclusion, we built a dataset, which can be used to experiment with and train future machine learning models for inverse gravity modeling. In order to incorporate additional information into the gravity inversion process, we modified the GeodesyNets introduced by Izzo and Gómez [6], and implemented a guidance loss, so that they are able to optimize for a specific density in a specified region inside of the body, in addition to the given gravitational field. We implemented the same loss for a more classical mascon based method and compared the results. We found that the mascon based method performed better than the GeodesyNet in most of the cases we studied, both regarding the gravitational field and how close the methods were able to get to the specified guidance density. Additionally, we identified some scenarios in which the GeodesyNet would be the better choice. To determine good guidance factors to scale the guidance loss for the different ground-truths, we performed an empirical search, and also experimented with automatically scaling the guidance loss. To mitigate the long training times of the GeodesyNet we tested post-training on an already trained model without guidance, which showed promising results in some situations.

REFERENCES

[1] A. Caldiero and S. Le Maistre. Small bodies global gravity inversion via the level-set method. *Icarus*, 411:115940, Mar. 2024. doi: 10.1016/j.icarus.2023.115940 1

[2] T. G. G. Chanut, S. Aljbaae, and V. Carruba. Mascon gravitation model using a shaped polyhedral source. *Monthly Notices of the Royal Astronomical Society*, 450(4):3742–3749, May 2015. doi: 10.1093/mnras/stv845 1

[3] G. Friege. 3d asteroid catalogue. 2

[4] A. Fujiwara, J. Kawaguchi, D. K. Yeomans, et al. The rubble-pile asteroid itokawa as observed by hayabusa. *Science*, 312(5778):1330–1334, June 2006. doi: 10.1126/science.1125841 1

[5] W. Heiskanen and H. Moritz. *Physical Geodesy*. 1, 4

[6] D. Izzo and P. Gómez. Geodesy of irregular small bodies via neural density fields. *Communications Engineering*, 1(1), Dec. 2022. doi: 10.1038/s44172-022-00050-3 1, 2, 3, 4, 5, 6, 7, 8

[7] A. Konopliv, R. Park, A. Vaughan, et al. The ceres gravity field, spin pole, rotation period and orbit from the dawn radiometric tracking and optical data. *Icarus*, 299:411–429, Jan. 2018. doi: 10.1016/j.icarus.2017.08.005 1

[8] D. S. Lauretta, S. S. Balram-Knutson, E. Beshore, et al. Osiris-rex: Sample return from asteroid (101955) bennu. *Space Science Reviews*, 212(1–2):925–984, Aug. 2017. doi: 10.1007/s11214-017-0405-1 1, 2

[9] H.-P. Li, R. Qi, J.-X. Hu, and Y.-X. Sun. 3d gravity anomaly inversion based on linknet. *Applied Geophysics*, 20(1):36–50, Mar. 2023. doi: 10.1007/s11770-023-1020-4 1, 2

[10] H. Meißenhelter, R. Weller, M. Noeker, et al. Adaptive polydisperse sphere packings for high accuracy computations of the gravitational field. In *2023 IEEE Aerospace Conference*. IEEE, Mar. 2023. doi: 10.1109/aero55745.2023.10115938 1

[11] J. Miller, A. Konopliv, P. Antreasian, et al. Determination of shape, gravity, and rotational state of asteroid 433 eros. *Icarus*, 155(1):3–17, Jan. 2002. doi: 10.1006/icar.2001.6753 1

[12] R. S. Park, R. A. Werner, and S. Bhaskaran. Estimating small-body gravity field from shape model and navigation data. *Journal of Guidance, Control, and Dynamics*, 33(1):212–221, 2010. doi: 10.2514/1.41585 1

[13] J. Pearl and D. Hitt. Comparing the computational efficiency of polyhedral and mascon gravity models. 02 2017. 1

[14] J. Pearl and D. Hitt. Mascon distribution techniques for asteroids and comets. *Celestial Mechanics and Dynamical Astronomy*, 134(6), Dec. 2022. doi: 10.1007/s10569-022-10115-2 1

[15] J. Sebera, A. Bezděk, I. Pešek, and T. Henych. Spheroidal models of the exterior gravitational field of asteroids bennu and castalia. *Icarus*, 272:70–79, July 2016. doi: 10.1016/j.icarus.2016.02.038 1

[16] H. Si. Tetgen, a delaunay-based quality tetrahedral mesh generator. *ACM Trans. Math. Softw.*, 41(2), Feb. 2015. doi: 10.1145/2629697 5

[17] L.-I. Sorsa, M. Takala, P. Bambach, et al. Tomographic inversion of gravity gradient field for a synthetic itokawa model. *Icarus*, 336:113425, Jan. 2020. doi: 10.1016/j.icarus.2019.113425 1

[18] Y. Takahashi and D. Scheeres. Morphology driven density distribution estimation for small bodies. *Icarus*, 233:179–193, May 2014. doi: 10.1016/j.icarus.2014.02.004 1

[19] J. Veverka, M. Robinson, P. Thomas, et al. Near at eros: Imaging and spectral results. *Science*, 289(5487):2088–2097, Sept. 2000. doi: 10.1126/science.289.5487.2088 1

[20] K. J. Walsh. Rubble pile asteroids. *Annual Review of Astronomy and Astrophysics*, 56(1):593–624, Sept. 2018. doi: 10.1146/annurev-astro-081817-052013 1

[21] R. A. Werner and D. J. Scheeres. Exterior gravitation of a polyhedron derived and compared with harmonic and mascon gravitation representations of asteroid 4769 castalia. *Celestial Mechanics and Dynamical Astronomy*, 65(3), 1997. doi: 10.1007/bf00053511 1

[22] P. T. Wittick and R. P. Russell. Mixed-model gravity representations for small celestial bodies using mascons and spherical harmonics. *Celestial Mechanics and Dynamical Astronomy*, 131(7), June 2019. doi: 10.1007/s10569-019-9904-6 2, 6

[23] G. Wu, Y. Wei, S. Dong, et al. Improved gravity inversion method based on deep learning with physical constraint and its application to the airborne gravity data in east antarctica. *Remote Sensing*, 15(20), 2023. doi: 10.3390/rs15204933 1, 2

[24] W. Yu-Feng, Z. Yu-Jie, F. Li-Hua, and L. Hong-Wei. Three-dimensional gravity inversion based on 3d u-net++. *Applied Geophysics*, 18(4):451–460, Dec. 2021. doi: 10.1007/s11770-021-0909-z

[1](#), [2](#)

- [25] F. Zhao, S. Xia, Z. Wu, et al. *Spherical U-Net on Cortical Surfaces: Methods and Applications*, p. 855–866. Springer International Publishing, 2019. doi: 10.1007/978-3-030-20351-1_67 [2](#)
- [26] Özgün Çiçek, A. Abdulkadir, S. S. Lienkamp, et al. 3d u-net: Learning dense volumetric segmentation from sparse annotation, 2016. [2](#)