



Fast Algorithms for Slow Moving Asteroids: Constraints on the Distribution of Kuiper Belt Objects

Peter J. Whidden¹, J. Bryce Kalmbach², Andrew J. Connolly¹, R. Lynne Jones¹, Hayden Smotherman¹, Dino Bektesevic¹, Colin Slater¹, Andrew C. Becker³, Željko Ivezić¹, Mario Jurić¹, Bryce Bolin¹, Joachim Moeyens¹, Francisco Förster^{4,5}, and V. Zach Golkhou^{1,6}

¹ Department of Astronomy, University of Washington, Seattle, WA 98195, USA

² Department of Physics, University of Washington, Seattle, WA 98195, USA; brycek@uw.edu

³ Amazon Web Services, Seattle, WA 98121, USA

⁴ Center for Mathematical Modeling, Beaucheff 851, 7th floor, Santiago, Chile

⁵ Millennium Institute of Astrophysics, Chile

⁶ The eScience Institute, University of Washington, Seattle, WA 98195, USA

Received 2018 August 28; revised 2018 December 31; accepted 2019 January 8; published 2019 February 15

Abstract

We introduce a new computational technique for searching for faint moving sources in astronomical images. Starting from a maximum-likelihood estimate for the probability of the detection of a source within a series of images, we develop a massively parallel algorithm for searching through candidate asteroid trajectories that utilizes graphics processing units (GPU). This technique can search over 10^{10} possible asteroid trajectories in stacks of the order of $10\text{--}15\ 4K \times 4K$ images in under a minute using a single consumer grade GPU. We apply this algorithm to data from the 2015 campaign of the High Cadence Transient Survey (HiTS) obtained with the Dark Energy Camera (DECam). We find 39 previously unknown Kuiper belt objects (KBOs) in the 150 square degrees of the survey. Comparing these asteroids to an existing model for the inclination distribution of the Kuiper belt we demonstrate that we recover a KBO population above our detection limit consistent with previous studies. Software used in this analysis is made available as an open source package.

Key words: methods: data analysis – minor planets, asteroids: general – Kuiper belt: general – techniques: image processing

1. Introduction

Traditional approaches for detecting trans-Neptunian objects (TNOs) rely on the identification of sources within individual images and then linking these sources to generate orbits (Kubica et al. 2007; Denneau et al. 2013). More recently digital tracking or shift-and-stack techniques have been developed to search for moving sources below the detection limit of any individual image (Gladman & Kavelaars 1997; Allen et al. 2001; Bernstein et al. 2004; Heinze et al. 2015). These approaches are fundamentally different from the traditional techniques in that they assume a trajectory for an asteroid and align a set of individual images along that trajectory in order to look for evidence for a source.

Shift-and-stack methods share many commonalities with the track-before-detect (TBD) method used for the tracking of satellites and missiles (e.g., Reed et al. 1988; Johnston & Krishnamurthy 2002). This field is mature in the literature and implementation, and has been generalized to enable the detection of not just linear motion, but also the tracking of acutely maneuvering non-cooperative targets (Rozovskii & Petrov 1999). We will adopt several of the features of TBD, in particular those described in Johnston & Krishnamurthy (2002), who outline the core principles of accumulating the track detection probability, and in quantifying the false alarm probability.

A related approach to faint moving object detection is presented in Lang et al. (2009), who describe a search for high proper motion stars. These objects move by a distance comparable to the point-spread function (PSF) FWHM over the course of an entire survey (i.e., years). Thus a direct image stack is sufficient to detect objects. However, Lang et al. (2009)

return to the individual science images to perform a joint fit for the proper motion and parallax of the objects, even though they appear at low signal-to-noise in any individual image. The scaling of detection depth in these techniques goes formally as $\Delta m = 1.26 \log(N)$ magnitudes, or 1 mag deeper after the linking of 6 faint detections, and 2 mag after 40 detections.

The advantage of digital tracking is that we increase the detection limit for a series of N images as \sqrt{N} (assuming a constant PSF and background across all of the images). For objects having power-law distributions in apparent magnitude—e.g., the double power-law TNO model of Bernstein et al. (2004)—linking six epochs of data would yield an increase of four to seven times as many objects from the same data. The cost of digital tracking comes from the combinatorial and computational complexity of having to search a large number of candidate trajectories for each pixel within an image. Digital tracking must combine the individual images along a proposed motion vector that will depend on the assumed distance of the asteroid. Even for slow moving asteroids searches will scale as Nn where n is the number of pixels in an image.

These computational costs have limited the application of digital tracking to searches for slowly moving objects or to narrow pencil-beam surveys. In this paper we introduce a new approach that utilizes a probabilistic formalism for the detection of sources in images (removing the need to stack or coadd images) and graphics processing units (GPUs) to massively parallelize the number of searches that can be undertaken concurrently. In Section 2 we introduce a maximum-likelihood formalism for the detection of sources and extend this for the case of moving objects. In Section 3 we apply this approach to the High Cadence Transient

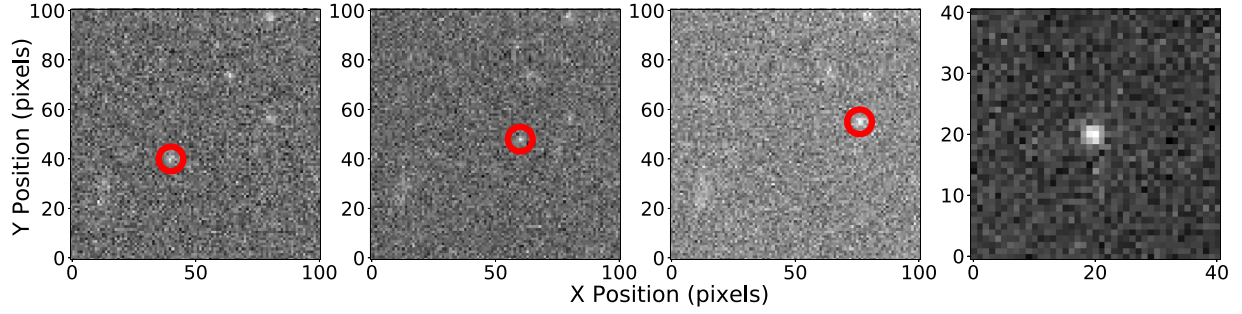


Figure 1. Shifting and stacking of individual images along the asteroid’s trajectory creates a single point source in the stacked image.

Survey (HiTS) and describe some of the filtering techniques that were applied to exclude false positives in the candidate asteroids. In Section 4 we discuss the asteroids detected by this approach and compare their properties to current models and observations.

2. Fast Tracking and Stacking of Images

Digital tracking assumes a set of N images have been observed over a period of time (from minutes to days) with the individual images covering approximately the same part of the sky. The individual images are astrometrically shifted along a proposed motion vector, coadded, and then searched for point sources in the resulting coadds (Gladman & Kavelaars 1997; Allen et al. 2001; Bernstein et al. 2004). This approach is illustrated in Figure 1 where the image on the far right is the sum of the previous three images added along the motion of the highlighted object. Creating a stack optimized for faint, moving objects must include astrometric offsets between the images that correspond to the distance that an object has moved between observations. Unfortunately, this angular velocity is not known a priori, and in fact differs for each moving object in the field. Thus a stacked search for solar system objects in time series data requires a sequence of coadds, each optimized for a particular motion vector. Due to the combinatorial complexity of the problem, these efforts have traditionally been optimized for TNO recovery, where apparent angular motions are small.

During an observation of a solar system object, there are apparent motion contributions from the object’s own space velocity, as well as from the reflex motion of the Earth, which is primarily due to the Earth’s motion around the Sun but also includes the Earth’s rotation around its axis. These contribute to an apparent angular motion of the solar system object, which will trail during an exposure. If the object trails by more than the PSF FWHM during observation, its signal is spread over additional background pixels, which lessens the overall signal-to-noise (Shao et al. 2014).

These trailing losses also apply to a stacked image: the image stack velocity must be close enough to the true velocity of the object to not spread the signal over more than the PSF FWHM. This requirement, together with the range of expected apparent motions of the desired objects, sets the number or sampling of the velocities (or orbits) that must be searched and stacks that must be examined. Typical apparent motions (at opposition) range from 20''/hr for main belt asteroids at 3 au to 1''/hr for TNOs.

2.1. A Likelihood-based Approach for Source Detection

2.1.1. Form of Single Pixel Likelihood Function

Our goal is to derive the likelihood functions for a given source present in the coaddition of a series of images. First we start by finding the form of the likelihood for a single pixel in a single image. The following derivation is based upon work in Kaiser (2004) and interpreted in Bosch (2015) and Bosch et al. (2018). Photons landing on a pixel in a detector follow Poisson statistics. For a given pixel the probability of counting n photons with an expected value of μ is defined as

$$P(n|\mu) = \frac{\mu^n e^{-\mu}}{n!}. \quad (1)$$

This probability can also be interpreted as the likelihood that a model of the sources within an image (hereafter the true image) generates a predicted count μ given we observe n counts on a pixel. The log-likelihood of our prediction model is, therefore,

$$\begin{aligned} \mathcal{L}(\text{model}) &= \ln P(\text{data}|\text{model}) \\ &= \ln P(n|\mu) = n \ln \mu - \mu - \ln n! \end{aligned} \quad (2)$$

Differentiating \mathcal{L} with respect to μ , the log-likelihood has its peak when $\mu_o = n$. Furthermore, if we Taylor expand around this maximum value then we get

$$\begin{aligned} \mathcal{L}(\text{model}) &= n \ln \mu_o - \mu_o - \ln n! \\ &+ \left(\frac{n}{\mu_o} - 1 \right) (\mu - \mu_o) - \frac{n}{2\mu_o^2} (\mu - \mu_o)^2 + \dots \end{aligned} \quad (3)$$

$$\mathcal{L}(\text{model}) = \text{constant} - \frac{1}{2} \frac{(n - \mu)^2}{n} + \dots, \quad (4)$$

where the constant contains all of the terms that depend only on n and so are independent of our model. Finally, if n is large we can ignore higher order terms and approximate our likelihood function as that of a Gaussian with likelihood proportional to $e^{-\frac{(n-\mu)^2}{2n}}$.

2.1.2. Pixels and PSF

At this point we need to take a step back and understand what exactly goes into calculating photon counts at each pixel. To do this we will follow the derivation laid out in Bosch (2015) but for a two-dimensional image. First, we start with the number of counts, $n(x, y)$, we get from the true sky intensity, $I(x, y)$, on a pixel centered at (x, y) . The observation on our

detector will be the true sky intensity convolved with the PSF, $T(x, y)$. In addition, there will be an extra integral to account for the binning into a pixel centered at (x, y) with side length a . That gives us

$$n(x, y) = \int_{x-a/2}^{x+a/2} \int_{y-a/2}^{y+a/2} dv du \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} dq dp \times I(p, q) T(u - p, v - q), \quad (5)$$

which can be rewritten in terms of two convolutions where we rewrite the pixel binning integral as a convolution with a square top hat function of $H(x, y)$ with height 1 and side length a

$$n(x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} dv du H(x - u, y - v) \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} dq dp \times I(p, q) T(u - p, v - q), \quad (6)$$

$$n(x, y) = [H * I * T](x, y), \quad (7)$$

and if we exploit the associative and commutative properties of convolutions we can rewrite this as

$$n(x, y) = [I * (H * T)](x, y) \quad (8)$$

and following the procedure in Bosch (2015) simplifying the term in parentheses to

$$T_a(x, y) = (H * T)(x, y) \quad (9)$$

and finally ending up with

$$n(x, y) = [I * T_a](x, y) \quad (10)$$

so that when we refer to the PSF below as T_a we are referring to the PSF including the pixel transfer function.

2.1.3. Full Likelihood Function for a Single Image

Now we understand how to represent the pixel data as a function of the sky and PSF as well as how to write out the likelihood function of a single pixel. We consider an image \mathbf{x} , where the i th pixel is \mathbf{x}_i and the real sky is modeled as $f(\mathbf{x})$. Then we use our assumption of a large photon count to justify a Gaussian likelihood function as described in Section 2.1.1 and get

$$\begin{aligned} L(\mathbf{x}_i) &= P(\text{data}|\text{model}) = P(n(\mathbf{x}_i)|f(\mathbf{x}_i)) \\ &= \frac{1}{\sqrt{2\pi\sigma_{\mathbf{x}_i}^2}} e^{-\frac{\frac{1}{2}(n(\mathbf{x}_i) - [f * T_a](\mathbf{x}_i))^2}{\sigma_{\mathbf{x}_i}^2}}, \end{aligned} \quad (11)$$

but if we are looking at the entire image plane we need to go from a single variable Gaussian to a multivariate Gaussian distribution. Thus, the likelihood for the full image is

$$\begin{aligned} L(\mathbf{x}) &= P(\mathbf{n}(\mathbf{x})|f(\mathbf{x})) \\ &= \frac{1}{\sqrt{(2\pi)^k |C|}} e^{-\frac{1}{2} \sum_{i,j} (n(\mathbf{x}_i) - [f * T_a](\mathbf{x}_i)) \times C^{-1}(\mathbf{x}_i, \mathbf{x}_j) \times (n(\mathbf{x}_j) - [f * T_a](\mathbf{x}_j))}, \end{aligned} \quad (12)$$

where k is the number of pixels and C is the pixel covariance matrix. If we take the log-likelihood function then we have

$$\begin{aligned} \mathcal{L}(\mathbf{x}) &= -\frac{k}{2} \ln(2\pi) - \frac{1}{2} \ln|C| - \frac{1}{2} \sum_{i,j} (n(\mathbf{x}_i) \\ &\quad - [f * T_a](\mathbf{x}_i)) \times C^{-1}(\mathbf{x}_i, \mathbf{x}_j) \\ &\quad \times (n(\mathbf{x}_j) - [f * T_a](\mathbf{x}_j)) \end{aligned} \quad (13)$$

for the full form of the log-likelihood of a given sky model $f(\mathbf{x})$.

2.1.4. Likelihood of Detection of a Point Source

If we want to find a point source at a pixel \mathbf{y} in the image with flux α then the sky model we are proposing is $f(\mathbf{x}) = \alpha \delta(\mathbf{x} - \mathbf{y})$, a delta function located at the pixel location of the source multiplied by the flux. Now let $n(\mathbf{x})$ represent the flux counts for each pixel we have in a background subtracted image and notice that the first two terms in Equation (13) are independent of the model $f(\mathbf{x})$. Furthermore, in order to keep values positive, we change to the negative log-likelihood function which we are now trying to minimize in order to get the most likely parameters. Thus, we get

$$\begin{aligned} \mathcal{L}(\alpha, \mathbf{y}) &= \text{constant} + \frac{1}{2} \sum_{i,j} (n(\mathbf{x}_i) - \alpha T(\mathbf{x}_i - \mathbf{y})) \\ &\quad \times C^{-1}(\mathbf{x}_i, \mathbf{x}_j) \times (n(\mathbf{x}_j) - \alpha T(\mathbf{x}_j - \mathbf{y})), \end{aligned} \quad (14)$$

where the constant holds terms that are model independent. Also, we have defined $\alpha T(\mathbf{x}_i - \mathbf{y}) = [f * T_a](\mathbf{x}_i)$, which is the convolution of the point-source sky model with the PSF and is thus equivalent to the PSF centered at \mathbf{y} . $T(\mathbf{x}_j - \mathbf{y})$ is the same for the pixels in the j indexed sum. We have two sums here since we are summing across every combination of pixels from the covariance matrix. If we multiply out these terms it looks like the following:

$$\begin{aligned} \mathcal{L}(\alpha, \mathbf{y}) &= \text{constant} + \frac{1}{2} \sum_{i,j} C^{-1}(\mathbf{x}_i, \mathbf{x}_j) (n(\mathbf{x}_i) n(\mathbf{x}_j)) \\ &\quad - \alpha \sum_{i,j} C^{-1}(\mathbf{x}_i, \mathbf{x}_j) n(\mathbf{x}_i) T(\mathbf{x}_j - \mathbf{y}) \\ &\quad + \frac{\alpha^2}{2} \sum_{i,j} C^{-1}(\mathbf{x}_i, \mathbf{x}_j) T(\mathbf{x}_i - \mathbf{y}) T(\mathbf{x}_j - \mathbf{y}). \end{aligned} \quad (15)$$

Furthermore we define the following terms

$$\Psi(\mathbf{y}) = \sum_{i,j} C^{-1}(\mathbf{x}_i, \mathbf{x}_j) n(\mathbf{x}_i) T(\mathbf{x}_j - \mathbf{y}) \quad (16)$$

$$\Phi(\mathbf{y}) = \sum_{i,j} C^{-1}(\mathbf{x}_i, \mathbf{x}_j) T(\mathbf{x}_i - \mathbf{y}) T(\mathbf{x}_j - \mathbf{y}) \quad (17)$$

and once again add into constant the terms that are not model dependent. Finally, this gives us the likelihood in a compact form of

$$\mathcal{L}(\alpha, \mathbf{y}) = \text{constant} - \alpha \Psi(\mathbf{y}) + \frac{\alpha^2}{2} \Phi(\mathbf{y}), \quad (18)$$

where $\Psi(\mathbf{y})$ and $\Phi(\mathbf{y})$ are new types of images that can be created using the PSF.

2.1.5. Coaddition of Likelihood Images for Point Source Detection

We make the assumption that the signal for the majority of the candidate detections will be dominated by the background noise and that the background noise is independent in each pixel. While there may be sources of correlation between pixels such as electronic detector effects (e.g., crosstalk or the dependence of the PSF on intensity), for this treatment we will assume that these effects are small enough to ignore and thus will continue as if we have a completely diagonal covariance

matrix. This simplifies our calculations into images that can be precomputed with a simple kernel that approximates the PSF.

Thus, our equations for the Ψ and Φ images are reduced to

$$\Psi(\mathbf{y}) = \sum_i \frac{1}{\sigma_i^2} n(\mathbf{x}_i) T(\mathbf{x}_i - \mathbf{y}) \quad (19)$$

$$\Phi(\mathbf{y}) = \sum_i \frac{1}{\sigma_i^2} T(\mathbf{x}_i - \mathbf{y})^2, \quad (20)$$

where Ψ is the inverse-variance weighted cross-correlation of the PSF and the data, which is also the convolution of the PSF rotated by 180° or just the PSF if it is symmetric. Φ is the effective area of the PSF weighted by the inverse variance (Bosch et al. 2018).

Bright pixels in the image can be dealt with by a mask and we set the inverse variance to zero so that they will add nothing to the likelihood sum. Since trajectories will only cross over this pixel once when we run them over a series of images this serves to give noisy pixels zero weight in the full sum of a trajectory while keeping the information in the other images.

Solving for the maximum-likelihood solution we end up with

$$\frac{\partial \mathcal{L}_{\text{ML}}}{\partial \alpha} = -\Psi(\mathbf{y}) + \alpha \Phi(\mathbf{y}) = 0 \quad (21)$$

and as a result we find that

$$\alpha_{\text{ML}} = \frac{\Psi(\mathbf{y})}{\Phi(\mathbf{y})} \quad (22)$$

and

$$\mathcal{L}_{\text{ML}}(\alpha_{\text{ML}}, \mathbf{y}) = \text{constant} - \frac{\Psi^2(\mathbf{y})}{2\Phi^2(\mathbf{y})}, \quad (23)$$

where α_{ML} is the most likely flux for a source at pixel \mathbf{x}_i and $\mathcal{L}_{\text{ML}}(\alpha_{\text{ML}}, \mathbf{y})$ is the probability of that source given the observation $n(\mathbf{x}_i)$. The kernel that maximizes the likelihood of our flux measurements is, therefore, the 180° rotation of the PSF or, in the case of a symmetric PSF, the PSF itself. Additionally, Equation (23) is the log form of a Gaussian—where by analogy $\Psi \simeq (x - \mu)$ and $\Phi^2 \simeq \sigma^2$ —so we can approximate a χ^2 distribution. If we define $\nu = \frac{\Psi}{\sqrt{\Phi}}$ and then make an image of ν , we can call points above some threshold value m to be m -sigma detections (Szalay et al. 1999).

In order to make a coadded likelihood image, we create our Ψ and Φ images from sums across all of the pixels in all of the images. This amounts to making Ψ and Φ likelihood images for each of our original images, i , with the appropriate PSF for each respective image and summing them separately so that

$$\Psi_{\text{coadd}} = \sum_i \Psi_i(\mathbf{y}_i), \quad (24)$$

$$\Phi_{\text{coadd}} = \sum_i \Phi_i(\mathbf{y}_i), \quad (25)$$

$$\nu_{\text{coadd}} = \frac{\Psi_{\text{coadd}}}{\sqrt{\Phi_{\text{coadd}}}}. \quad (26)$$

Here, ν is the signal-to-noise of the detection of a source at pixel \mathbf{x} within the coadded image. Therefore, the final value for the likelihood of a point object is just the sum of the values at a given set of points in the Ψ image divided by the sum of the

values at the same points in the Φ image. This means that if we wish to do a moving object detection, all that is needed is to sum the Ψ and Φ values as shown, but we must use the appropriate pixel coordinates for a given trajectory as the \mathbf{y}_i values in each image—there is no need to shift and stack the likelihood images at any point. The Ψ and Φ images can also be precalculated meaning we are able to store them in memory as many trajectories are searched.

2.2. Object Detection as Optimization

Detecting moving point sources in a stack of images can now be approached as an optimization problem. For the linear trajectories we are considering, the core task can be reduced to finding the initial positions \mathbf{y}_0 and velocities \mathbf{v} so that ν is maximized. For a given potential object, the best candidate source is then

$$\text{argmax}_{\mathbf{y}_0, \mathbf{v}} (\nu_{\text{coadd}}). \quad (27)$$

We hope to find all unique \mathbf{y}_0 and \mathbf{v} such that $\nu(\mathbf{y}_0, \mathbf{v})$ is above a desired detection threshold. The most straightforward computational approach to this problem is to evaluate ν for all possible values of \mathbf{y}_0 and \mathbf{v} that describe realistic orbits. An illustration of this approach is shown in Figure 2 starting from a single value of \mathbf{y}_0 . This method directly computes Ψ and Φ for every relevant trajectory through all of the images by sampling pixels along each trajectory and storing the result if the integrated ν is above the threshold. The computational complexity of this algorithm is bounded by $\mathcal{O}(na(tu)^2/p)$ where n is the number of images to be stacked, a is the area on the sky to be searched, p is the area of a single pixel, t is the duration between the first and last image, and u is the range of an object's apparent velocity. The complexity scales with time and velocity range quadratically because all trajectories ending anywhere inside an area with radius tu must be considered.

2.3. GPU Implementation

Comprehensively searching a stack of images for moving objects directly requires computing the value of ν billions of times. Fortunately each evaluation of ν is entirely independent from the others, and this allows for natural parallel execution. Rectangular patches of adjacent trajectories are grouped into thread blocks with dimensions of 32×16 which are distributed across the GPU's multiprocessors. Ψ and Φ pixels are interleaved in memory and within each thread block horizontally adjacent trajectories access them contiguously, enabling high throughput. To achieve good performance all of the images must be stored in the GPU's memory at once, which limits the size and number of images that can be used to about $100 \text{ K} \times 4 \text{ K}$ images. In general, the run time of this algorithm can be estimated as Nnk where N is the number of images in the stack, n is the number of trajectories, and k is a performance constant that is experimentally determined by the computer hardware and implementation. In our implementation we have a measured value of k about 2.4×10^{-11} s per image for each trajectory. Practically this means searching for a wide range of objects (240 billion trajectories) in a stack of 30 images ($4 \text{ K} \times 4 \text{ K}$) takes about 180 s. Besides computing ν , we also use the GPU for convolving the images. The convolution is done in a single pass with a spatially invariant kernel. In this work we only used a Gaussian PSF approximation, and could

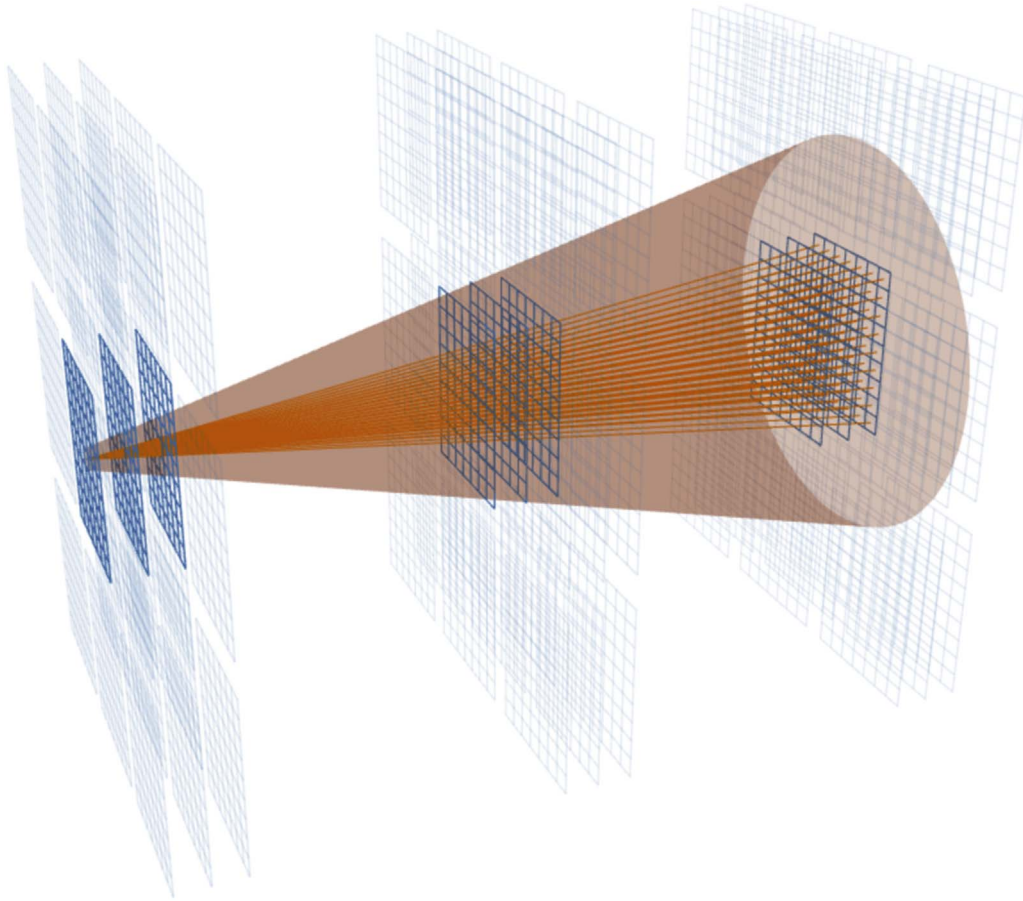


Figure 2. Visualization of the many trajectories that must be searched in order to cover a defined velocity and angle range over a stack of images of the same field taken at different times.

have used a two-pass separable convolution but chose to use a slower single pass to support non-Gaussian PSFs in the future.

3. Searching for Faint KBOs

3.1. The High Cadence Transient Survey

We tested our software using the first three nights (2015 February 17–19) of the 2015 campaign of HiTS (Förster et al. 2016). The 2015 HiTS campaign involved repeatedly visiting 50 three-square-degree fields in the DECam *g*-band filter. There were typically 5 visits per night to each of the 50 fields, taken with a 1.6 hr cadence. The remaining three nights of the 2015 HiTS campaign data, also taken with the DECam *g*-band, were used for follow up of the detected objects (Förster et al. 2016).

These data were taken using the Dark Energy Camera (DECam) at Cerro Tololo Interamerican Observatory (CTIO). The best quartile of seeing at CTIO is about $0''.4$ FWHM. DECam has 60 $2K \times 4K$ CCDs, each with a pixel scale of 0.26 arcsec/pixel. We processed all of the data using the Large Synoptic Survey Telescope (LSST) Data Management (DM) Software (Jurić et al. 2015) and ran our software using the warped science images that were laid out in $4K \times 4K$ pixel patches. This means that the effective field of view of each warped image is about 0.34 square degrees (Flaugher et al. 2015). Because we currently do not search trajectories across CCDs, this is the effective field of view of any individual

search. We also used the masks and variance planes from the LSST DM processing output.

3.2. Application of the KBO Search

We created Ψ and Φ images from the individual science images. To mask static objects we identify all of the sources that are detected at more than 5σ above the background and mask the pixels associated with these sources. Masks for each static source are grown by an additional two pixels (in radius) to exclude lower surface brightness halos around these sources. To ensure that we do not mask bright *moving* sources we require that any masked pixel in a science image must be masked in at least one of the other science images (i.e., a source must be present at the same position in two of the images for it to be defined as static and masked). We apply the union of the mask for all individual images as a global mask to the final science images.

Our search started at every pixel in the earliest observation (in time) for each HiTS field and covered a range of 32,000 trajectories across the stacks of 12–13 images corresponding to three nights of HiTS data. The 32,000 trajectories came from a linearly spaced grid of 128 angular steps within $\pm 12^\circ$ of the ecliptic and 250 steps in velocities ranging from $1''$ – $5''/7$ /hr. This grid was set up so that at the end of the search period, our maximum separation between the final pixels in a search pattern would be no more than approximately 2 PSF widths. This means that that we would be within 1 PSF width of any

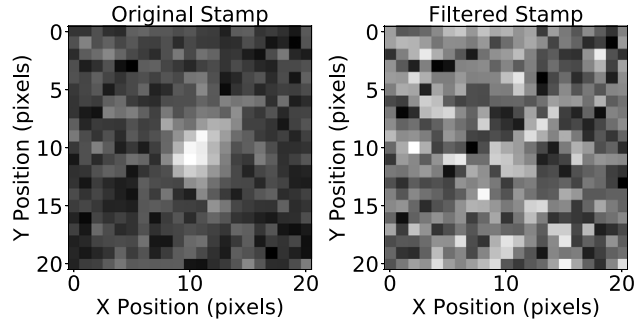
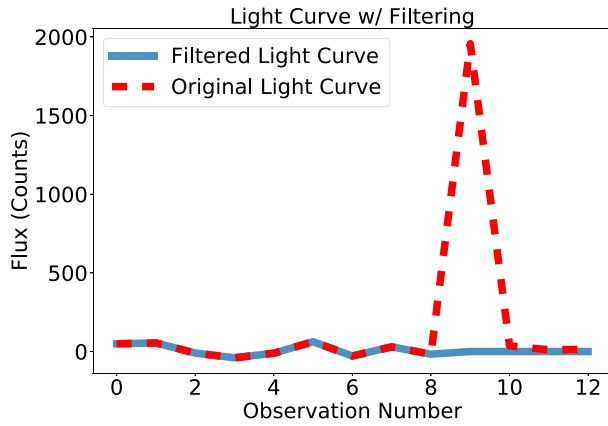


Figure 3. Left: change in light curve when an image with an outlier flux is removed from the light curve. Right: shifted and stacked postage stamps before and after outlier removal. After the single outlier observation is removed by the filter the trajectory is obviously not following a true object and is discarded.

possible trajectory. For this search, we went down to a signal-to-noise ratio (S/N) threshold of 10, as measured by the ν parameter introduced in Section 2.1.5. For these data, this corresponds to a single image limiting magnitude in g of 23.1.

3.3. Filtering of Candidate Trajectories

After running the initial GPU search, we use a set of Python tools to filter out false detections. One such tool is an outlier filtering process that modifies estimates of the flux and variance of an object given a sequence of observations. This is done using the light curve of the candidate trajectory and a variation on the Kalman filter (Kalman 1960) commonly used in signal processing. We filter out any observations that are more than five standard deviations away from the best-fit Kalman flux at that observation and use the remaining observations to recalculate the likelihood of the candidate. As an example of this process, Figure 3 shows postage stamps of the candidate before and after the filtering. In this case a fast, bright, moving object moved across the trajectory in a single image. Excluding that image leads us to the correct decision to discard this candidate trajectory, as the likelihood of an object along this trajectory drops to near zero, once the interloping object is removed.

After the outlier detection we build coadded postage stamps for all remaining candidate trajectories. We then use scikit-image (van der Walt et al. 2014) to calculate the central moments of the images and filter based upon the similarity of the moments to a Gaussian centered at the middle of the stamp. This does a good job of eliminating elongated shapes and trajectories where a bright source appeared in a single image but was off center. For example, in Figure 4 the left column shows postage stamps of real objects that passed through the filtering. The right column of the figure shows postage stamps that made it through the outlier filtering but were ruled out after the image moment filter.

Our final step in the filtering process is clustering to remove duplicate results from the same object. We take the starting x and y pixels and the horizontal and vertical velocities of the candidate trajectories and use the DBSCAN clustering method (Ester et al. 1996) in the scikit-learn python package (Pedregosa et al. 2012) to group similar trajectories. We then take the highest likelihood trajectory for each group and save the results with postage stamps and light curves to file for final examination by eye.

3.4. Found Objects

3.4.1. KBO Properties in the HiTS Field

In total we found 45 KBOs in our search, of which only 6 were previously detected by the Pan-STARRS 1 (PS1) survey according to the Minor Planet Center (MPC). PS1 used the Pan-STARRS Moving Object Processing System which links detections from sources identified in individual difference images (Denneau et al. 2013).

The full information for all our object detections is shown in Table 1. We used the orbit fitting code of Bernstein & Khushalani (2000) for the initial orbit determination and used the remaining HiTS data (see Section 3.1) to get additional observations where possible before submitting to the MPC. From this we estimated that, for the 45 KBOs, semimajor axes ranged from 21 to 67 au and DECam g -band magnitudes ranged from 22.1 to 24.7 mag. Figure 5 compares the semimajor axis to inclination for our discoveries and shows the overlap with KBO populations commonly discussed in the literature.

3.4.2. Comparison with Known Asteroids

The candidate objects were checked with the MPC database of known objects. Of the 45 objects that were detected, 6 were known. Four of these objects were detected in our search down to an S/N threshold of 10 and they are noted in Table 1. The 2 remaining objects were much fainter at V magnitudes of 24.1 and 24.3 while the faintest MPC object we recovered in our search was at $V = 23.0$. We attempted to recover the objects at a fainter S/N threshold by rerunning the search on the image stacks corresponding to the predicted locations. One of the objects was recovered when we reduced our S/N threshold to 5.96. The final object ran from the edge of one image stack to another between the first and second night. Our code cannot currently account for this situation in our search. However, we ran the search using only the second and third nights of data to find the object and our code was able to find it at a reduced S/N value of 5.31. This means that when going to a faint enough search threshold we were able to recover all of the known KBOs in the search area.

After submission to the MPC database two more results were matched to objects in the MPC catalogs. Previous observations had not predicted the orbits to fall within the observation fields, but were linked and recalculated by the MPC after submission

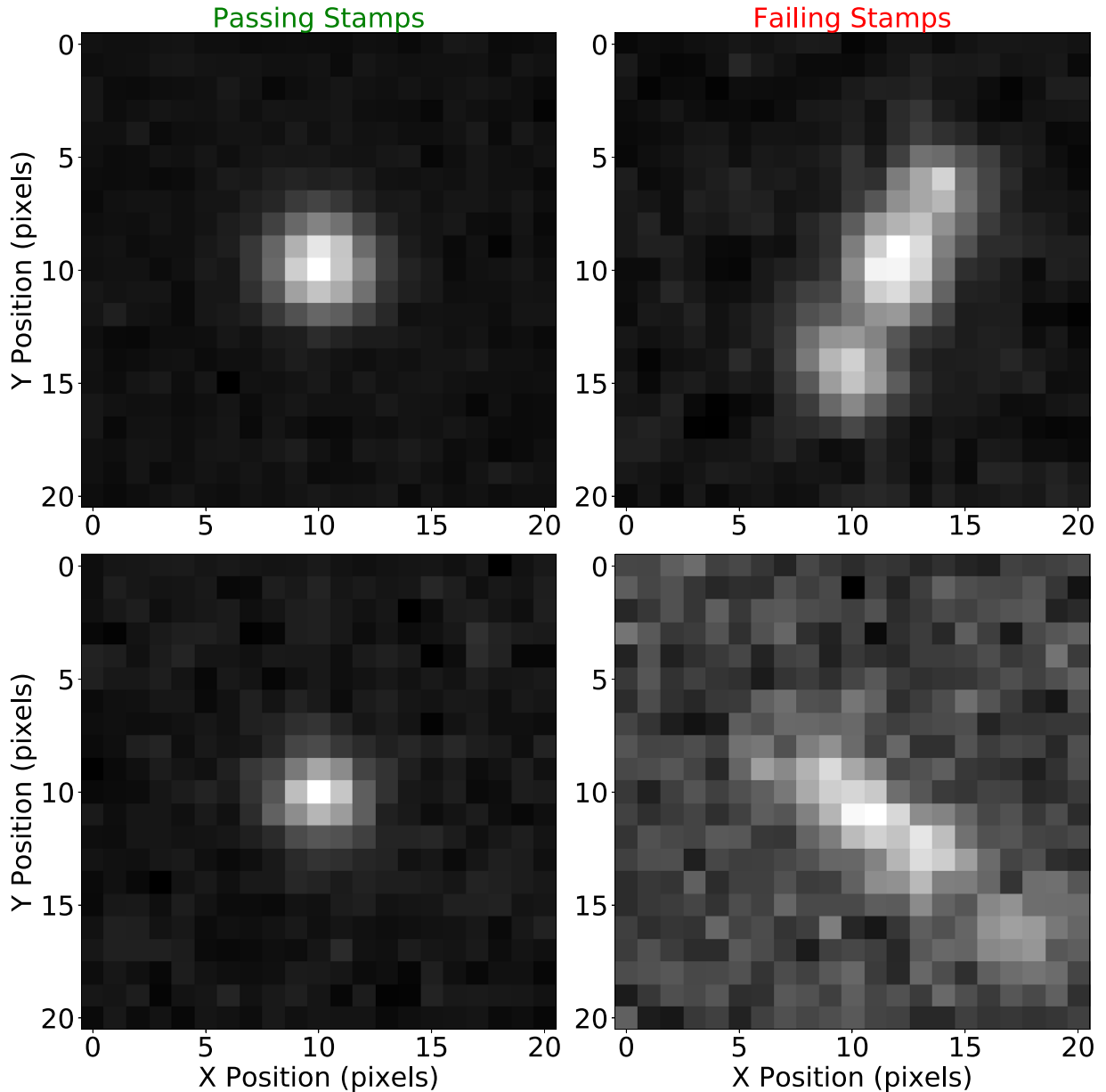


Figure 4. Left: stamps that passed the light curve filter and the image moment filter. Right: stamps that passed the light curve filter but were rejected by the image moment filter.

of our results. These new matches are also noted as previously discovered in Table 1.

4. Results

4.1. Recovery Efficiency

To understand the efficiencies of our search method, we inserted simulated objects with g magnitudes between 20 and 26 into the images from one of the 2015 HiTS fields and tested our ability to recover these objects. The objects all had a simple Gaussian PSF that matched our search PSF and the velocity angles and magnitudes were uniformly distributed within the ranges of our search parameters. Figure 6 shows the results in terms of counts on the left and the fraction of the total simulated set on the right. Both plots are a function of DECam g magnitude with bin widths of 0.25 mag. For the right plot we

fit an efficiency function of the form $f(m) = f_0 / e^{1 + \frac{m-L}{w}}$, where f_0 is the efficiency ceiling, L is the 50% detection probability magnitude, and w is the width in magnitudes of the drop-off in sensitivity.

>We plotted two efficiency functions, one for the entire range of magnitudes of the simulated objects and then one at magnitudes of $g > 21$. The drop-off at the brighter magnitudes is due to the extra masking we did at a specific count threshold. We will say more about this effect in the discussion in Section 5. On the fainter end we see the limit set by the S/N cutoff at $S/N = 10$. The detection efficiency ceiling for the full range is $84.6 \pm 3.7\%$ with a 50% threshold at $g = 24.13 \pm 0.07$ and drop-off width of 0.11 ± 0.07 mag. When we look at efficiencies at $g > 21$ we get an efficiency ceiling of $92.5 \pm 1.2\%$ and a 50% detection threshold at the very similar $g = 24.10 \pm 0.02$ value and drop-off width of 0.14 ± 0.02 mag.

Table 1
Detected KBOs in the HiTS Field with the Estimated Orbit Properties from the HiTS Data

MPC Designation	Semimajor Axis (au)	Eccentricity	Inclination	g_mag	New Discovery?
2015 DZ248	67.33	0.42	17.60	23.68	Yes
2015 DT248	21.38	0.36	22.21	24.14	Yes
2015 DP249	43.67	0.20	25.76	23.74	Yes
2015 DY248	48.02	0.28	14.72	24.19	Yes
2015 DX248	37.61	0.02	24.67	24.24	Yes
2015 DV248	42.32	0.67	27.67	23.94	Yes
2015 DW248	48.52	0.62	31.42	23.02	Yes
2015 DU248	44.24	0.02	14.44	23.99	Yes
2015 DQ248	63.15	0.45	33.13	24.05	Yes
2015 DR248	41.61	0.31	15.81	23.95	Yes
2015 DS248	42.67	0.78	23.02	24.00	Yes
2015 DO248	46.90	0.27	22.65	24.08	Yes
2015 DP248	48.97	0.70	21.73	24.15	Yes
2015 DO249	48.99	0.29	19.88	24.35	Yes
2015 DP249	49.49	0.43	17.06	24.28	Yes
2015 DZ249	45.05	0.25	14.27	24.30	Yes
2015 DY249	38.14	0.03	25.72	24.41	Yes
2015 DN249	38.30	0.03	17.08	24.34	Yes
2015 DM249	44.39	0.14	12.05	23.74	Yes
2015 DX249	47.08	0.24	30.04	23.22	Yes
2015 DL249	44.18	0.27	13.83	24.23	Yes
2015 DK249	44.29	0.02	7.03	24.05	Yes
2015 DH249	46.85	0.02	20.71	24.63	Yes
2015 DJ249	42.81	0.19	12.99	23.44	Yes
2015 DW249	38.83	0.19	15.11	23.68	Yes
2015 DC249	41.67	0.16	20.77	23.62	Yes
2015 DD249	65.74	0.32	19.25	23.65	Yes
2015 DB249	40.31	0.27	24.58	23.56	Yes
2015 DU249	59.50	0.21	18.87	24.72	Yes
2015 DA249	38.40	0.15	19.97	23.61	Yes
2015 DF249	45.35	0.44	12.99	24.08	Yes
2014 BV64	50.20	0.31	15.43	22.10	No
2015 DE249	41.66	0.35	20.29	23.35	Yes
2015 DV249	54.12	0.15	11.79	24.00	Yes
2015 BF519	45.12	0.39	18.42	23.00	No
2015 DG249	64.80	0.33	15.44	24.19	Yes
2011 CX119	47.70	0.46	29.47	23.19	No
2015 DA250	45.12	0.29	34.11	23.89	Yes
2015 DB250	44.13	0.27	10.99	23.80	Yes
2014 XW40	58.03	0.32	16.43	23.39	No
2015 DQ249	28.07	0.44	43.91	23.93	Yes
2015 DR249	41.10	0.02	35.13	23.81	Yes
2014 XP40	78.02	0.62	11.53	22.33	No
2013 FZ27	57.12	0.25	13.54	22.73	No
2015 DS249	31.08	0.29	11.38	24.37	Yes

4.2. Comparison with Existing Models

While our original searches used only three nights of data, we had additional nights in the HiTS data that we used for supplying additional observations when estimating the orbits of the discovered KBOs. This still limited our baseline for the discovered objects to a range from 3 to 10 days and made accurate estimates of semimajor axis and eccentricity difficult. We have confident measurements on the inclinations and all our inclination estimates for objects in the MPC catalog matched the published inclinations within 1σ of the output values from the orbit fitting code of Bernstein & Khushalani (2000). Therefore, we present a comparison of our inclination distribution to the general KBO population such as that in Brown (2001).

4.2.1. Inclination Distribution

Our inclination values range from 7° to 44° where the lower limit comes from the fact that our closest field to the ecliptic is at -6.4° . Even though our fields are all at moderate latitudes off the ecliptic (as far as -21.3°), Brown (2001) provides a method to compare our results to a predicted distribution using the inclinations of the objects and the ecliptic latitude of discovery.

Brown (2001) estimates an inclination distribution for the full KBO population with a double Gaussian multiplied by $\sin i$,

$$f_i(i) = \sin i \left[a \exp\left(\frac{-i^2}{2\sigma_1^2}\right) + (1 - a) \exp\left(\frac{-i^2}{2\sigma_2^2}\right) \right], \quad (28)$$

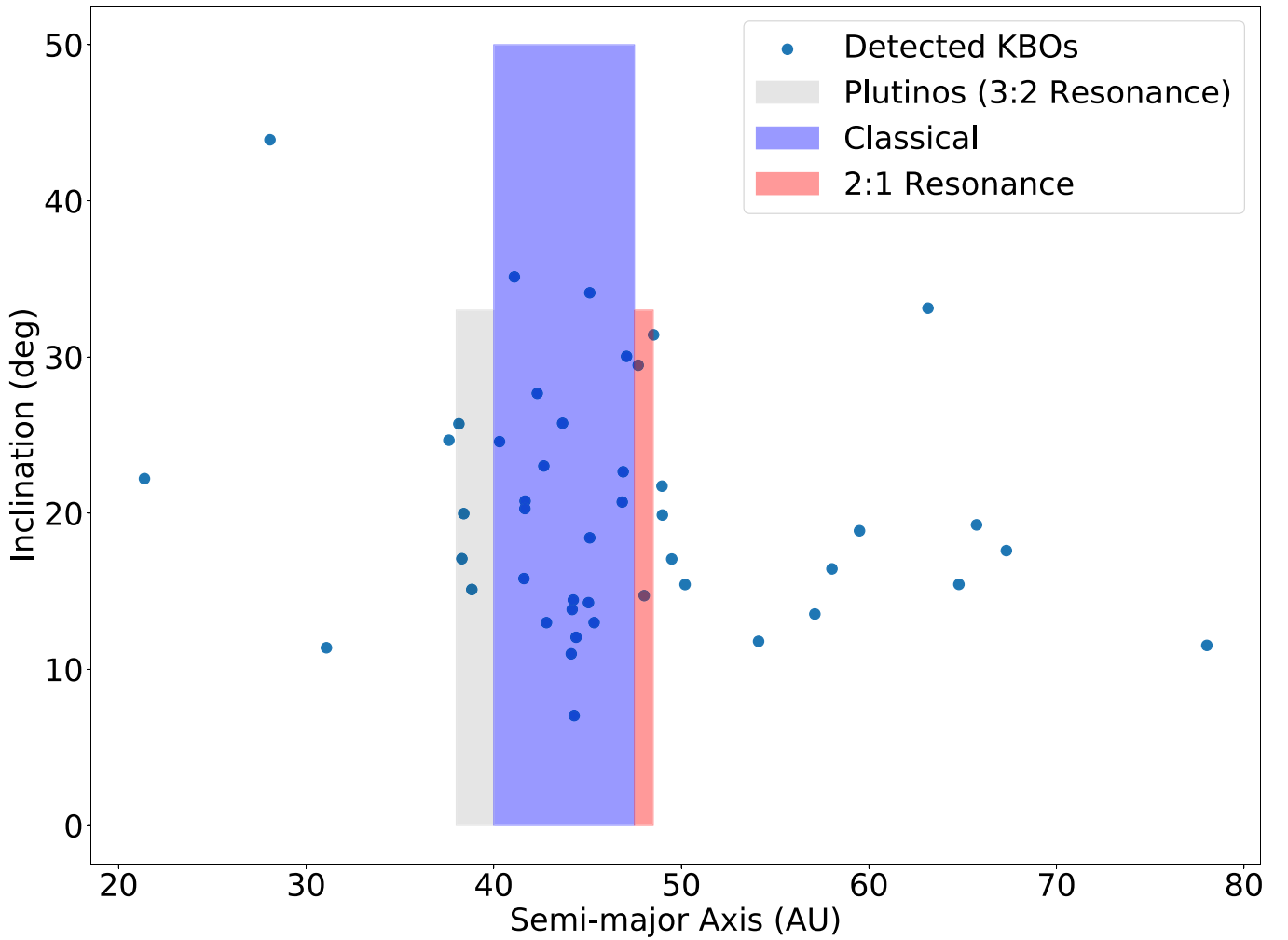


Figure 5. Semimajor axis vs. inclination for detected objects with the range of different KBO populations highlighted.

where $a = 0.89$, $\sigma_1 = 2^\circ.7$, and $\sigma_2 = 13^\circ.2$. To compare our results to this distribution we follow the method outlined in Section 3 of Brown (2001). For a given inclination distribution, f_i , the probability that an object, j , with discovery latitude, β_j , would have an inclination equal to or below the actual inclination, i_j , given by

$$P_j = \int_{\beta_j}^{i_j} \frac{f_i(i')}{(\sin^2 i' - \sin^2 \beta_j)^{1/2}} di' \times \left[\int_{\beta_j}^{\pi/2} \frac{f_i(i')}{(\sin^2 i' - \sin^2 \beta_j)^{1/2}} di' \right]. \quad (29)$$

The distribution of P_j for the actual KBO population estimated by Brown (2001) varies uniformly between 0 and 1. Therefore, if we take as the null hypothesis for our observations that they are an unbiased sample down to our magnitude limits and representative of the distribution of KBO inclinations in the fields we searched we should compare the distributions of P_j for our objects to the uniform distribution. To do this we start by calculating P_j for each of our objects and plot their sorted distribution in Figure 7. We compare our observed distribution to the inclination distribution of Brown (2001) using the Kuiper variant of the Kolmogorov–Smirnov (K-S) test as done in

Brown (2001) and according to the following equation:

$$D = \max(P_j - j/N). \quad (30)$$

The actual test statistic is $D\sqrt{N}$ where N is the sample size which is $N = 45$ in our data set. In order to find the confidence levels for the test, we create 10^5 sets of $N = 45$ random samples drawn from a uniform distribution and calculate $D\sqrt{N}$ comparing to a uniform distribution. The 1σ confidence value occurs when the probability of getting higher than a given $D\sqrt{N}$ value is 84.1%. We find this to be at $D\sqrt{N} = 1.47$.

Finally, we calculate the P_j values using the Monte Carlo methods described in Brown (2001). We first draw 10^5 inclinations from the Brown (2001) distribution and randomly place them along circular orbits. For each of our observed objects, j , we then take all of the Monte Carlo objects within $\pm 0.5^\circ$ of the latitude of discovery, β_j , and construct an empirical inclination distribution. In order to derive P_j for an object, we use this distribution to calculate the probability that an object with the given β_j will have an inclination at or below i_j . Using this set of P_j values we then perform the K-S test compared to a uniform distribution between 0 and 1. We perform the Monte Carlo simulation 1000 times and use the mean $D\sqrt{N}$ value as our test statistic for comparison. The K-S test comparison for one of the Monte Carlo distributions is shown in Figure 7. Our mean $D\sqrt{N}$ value after 1000 runs was

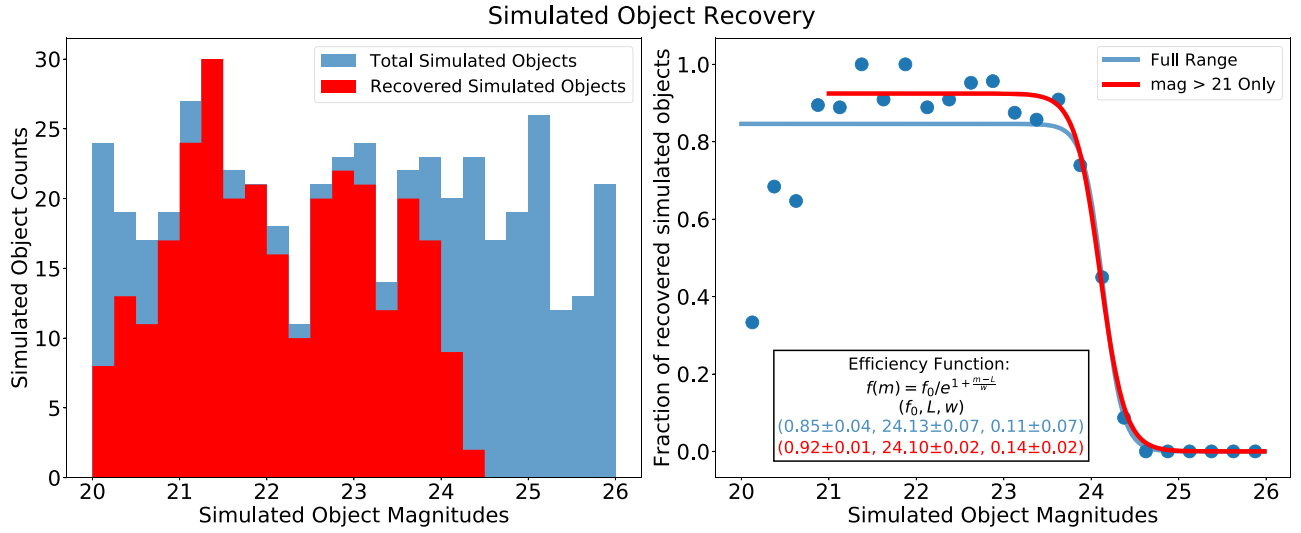


Figure 6. Recovery of simulated objects inserted into a HiTS field. Left: histogram comparing counts of recovered simulated objects to the full set as a function of magnitude. Right: fraction of recovered simulated objects as a function of magnitude fitted with an efficiency curve for both the full range of simulated objects and for $g > 21$ only.

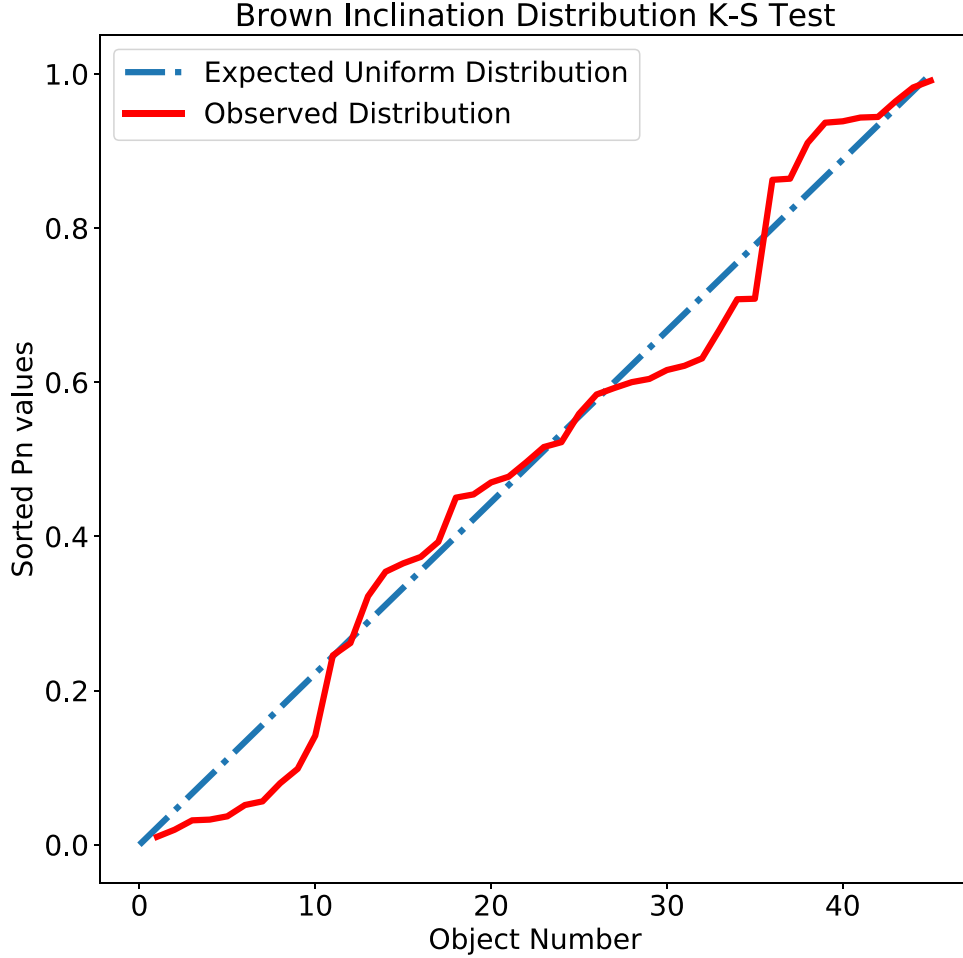


Figure 7. Kuiper variant of K-S Test comparing the inclination distribution of Brown (2001) to our recovered results.

1.37, corresponding to a confidence level of 75% and within the 1σ level. This means that we cannot reject the hypothesis that our observations come from the Brown (2001) distribution and are consistent with this prediction.

As a further comparison we did a basic survey simulation to determine the expected distribution of objects we would find in the HiTS data. We used the Monte Carlo distribution of inclinations and locations of objects scattered around circular

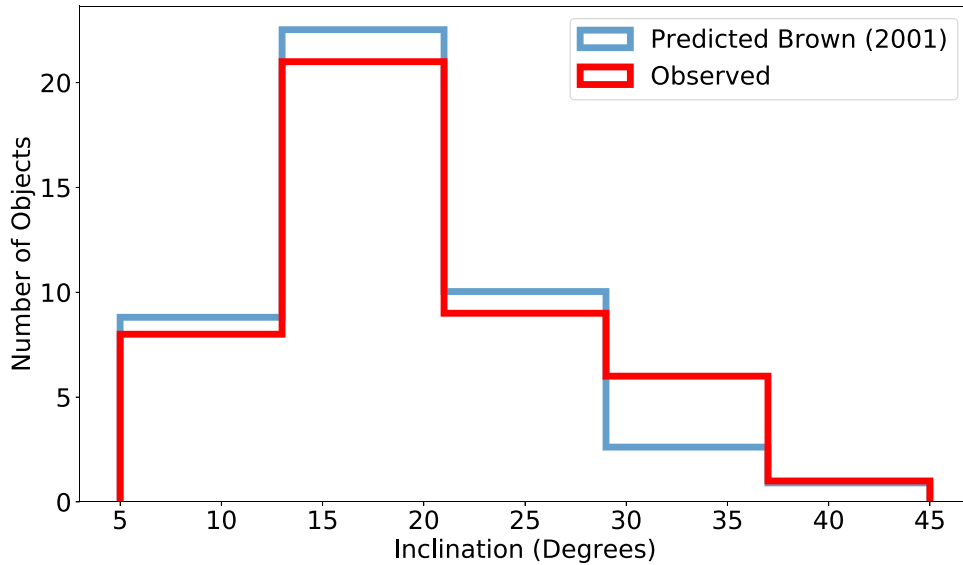


Figure 8. Inclination distributions of detected objects in the HiTS fields compared to predicted Brown distribution accounting for the ecliptic latitudes of the HiTS observations and normalized to the same number of discovered objects.

orbits from Figure 7 along with the locations of the HiTS fields and approximated the DECam field of view as a circle with $2^\circ.2$ diameter. With this information, we recorded the objects in the Monte Carlo simulation visible through the survey pattern. We then scaled this distribution to the discovery of the same number of objects that we found in the data and plotted them on top of one another in Figure 8. A χ^2 test on this data gives $\chi^2 = 2.23$ and a p -value of 0.69, meaning we cannot reject the hypothesis that our observed samples come from the Brown (2001) inclination distribution function for the general KBO population.

4.2.2. Magnitude Distribution

We next compare the magnitudes of our discoveries to the KBO magnitude distribution of Fraser et al. (2008). This distribution gives the number of observed objects per square degree as

$$N_{\text{obs}} = 10^{\alpha(m-m_o)}, \quad (31)$$

where $\alpha = 0.65 \pm 0.05$ and $m_o = 23.42 \pm 0.13$ for R magnitudes. Since Equation (31) is based upon the number of expected objects per square degree at the ecliptic we needed to scale our viewing area appropriately. Satisfied by the work in Section 4.2.1, where we showed that our results are consistent with the Brown (2001) inclination distribution, we converted this to a latitudinal distribution. We then multiplied the three-square-degree DECam viewing area for each field by the fraction of expected objects at the field’s ecliptic latitude compared to the maximum value at the ecliptic. This gave us an effective viewing area of 91.05 square degrees. The next step was converting between the R magnitudes used for Equation (31) and the g magnitudes of our observations. Fraser et al. (2008) also had to do magnitude conversions for various data sets and used a KBO $\langle g' - R \rangle$ color of 0.95 which we use here as well. Putting together the scaling and magnitude offsets we compare our results to the Fraser et al. (2008) expected number of objects for our survey fields in Figure 9. The drop-off expected from Section 4.1 around the 24th magnitude is

present and seems to occur after $g = 24.25$ after which we fall below 50% completeness which is consistent with the magnitude where our search drops below 50% efficiency as shown in Section 4.1. Figure 9 also shows the 10σ threshold we used in our search of the HiTS data. A χ^2 test on the data for $g < 24.25$ gives $\chi^2 = 10.21$ and a p -value of 0.25, meaning our results are consistent with the Fraser et al. (2008) luminosity distribution.

5. Discussion

5.1. Filtering Analysis

We used the field with simulated objects from Section 4.1 to study the effects of the various stages of our filtering process. The red line in Figure 10 shows the results for the searches over the full field in our processing which included an extra masking step that we discuss below. We start with the total number of searches based upon the number of grid steps multiplied by the number of pixels on the focal plane. We only keep those above our detection threshold which is the biggest single reduction in the number of possible results. After that, each step in our filtering process is able to reduce the number of false positives by 1–3 orders of magnitude in the number of the total results. The most effective is the postage stamp filtering which uses the moments of the postage stamp image and their resemblance to those of a Gaussian source model.

5.2. Masking and Threshold Effects

We made a series of decisions in our search based upon our target objects and the use of science images instead of less contaminated difference images. The first choice we made was to include an additional bright pixel mask when creating our Ψ and Φ images. This mask was in addition to the masking described in Section 3. Any pixel in any image that exceeded 120 counts, corresponding to $g \sim 21$, was masked in our searches. This additional masking covers extended bright halos, that were not covered by our initial footprint masking, and bright fast moving objects which are likely much closer to Earth than our target population. The effects of additional

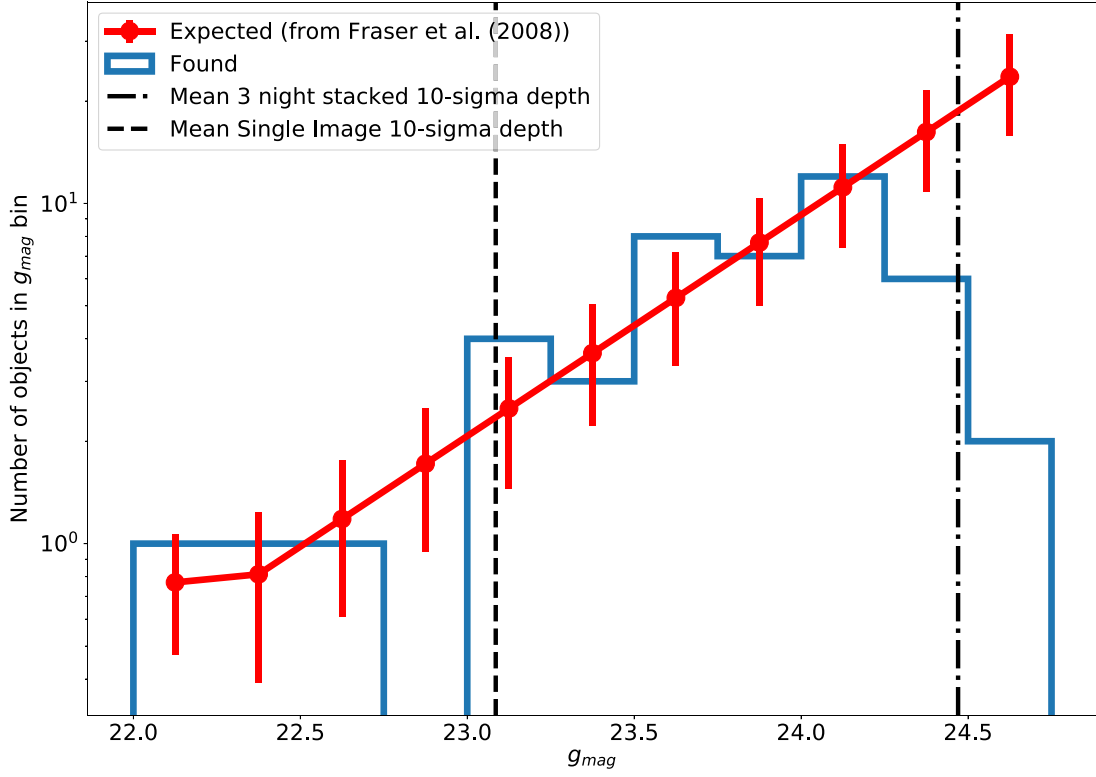


Figure 9. Completeness comparison of KBOs found in the HiTS survey using Kernel Based Moving Object Detection (KBMOD). Our results are consistent with a complete sample at the 24th magnitude compared to the single image depth of 23.1.

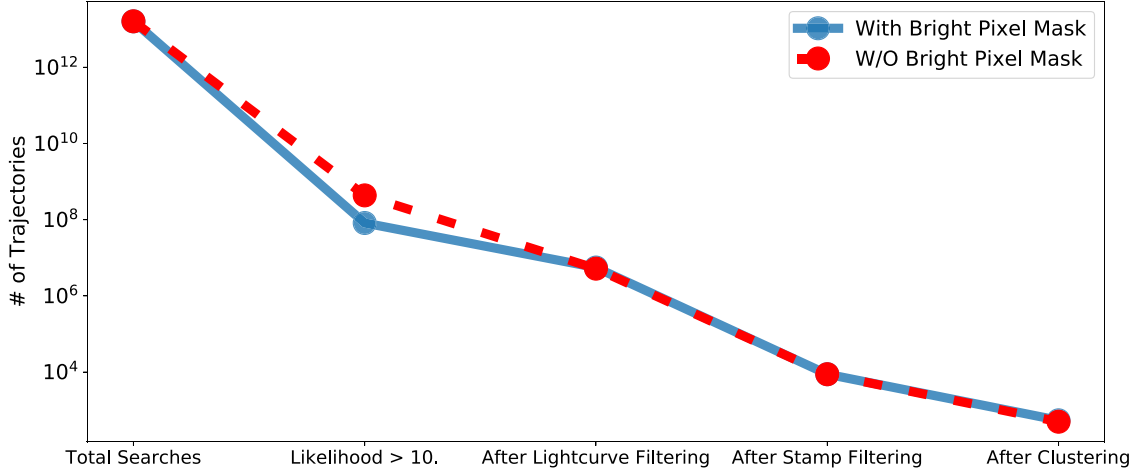


Figure 10. Number of positive trajectories after each step of filtering the search results from a field with simulated objects inserted.

masking on recovery were tested on the simulated object field (Section 4.1). Comparing the red line with this extra masking to the blue line without it in Figure 10 shows the major benefit of this masking was a significant decrease in the amount of false positives we had to filter out after our likelihood cut. The amount of positive results after the likelihood cut in Figure 10 shows about five times more objects to filter after the threshold cut without the masking. A decrease of 4×10^8 false positives over a single field saves us hours of computation time during our analysis since we processed the results through the light curve filter at a rate of 70 s for 500,000 objects. Over 50 fields this adds up to days of processing time that we were able to avoid.

However, we also looked to see what was the price we paid for this extra computational efficiency. Figure 11 compares the efficiency curves of Section 4.1 and Figure 6 to the results without the additional bright pixel masking. The recovery efficiency is higher across all magnitudes at $92.7 \pm 1.5\%$, but when we compare to the recovery at only $g > 21$ with the masking it is very similar to the $92.4 \pm 1.2\%$ of that result. The 50% efficiency depth is nearly unchanged at $g = 24.09 \pm .03$ without the masking meaning the bright pixel masking does not affect our overall depth but only the bright end of our recovery at $g < 21$. Using the Fraser et al. (2008) curve for the expected number of KBOs for magnitudes $20 < g < 21$ in our survey

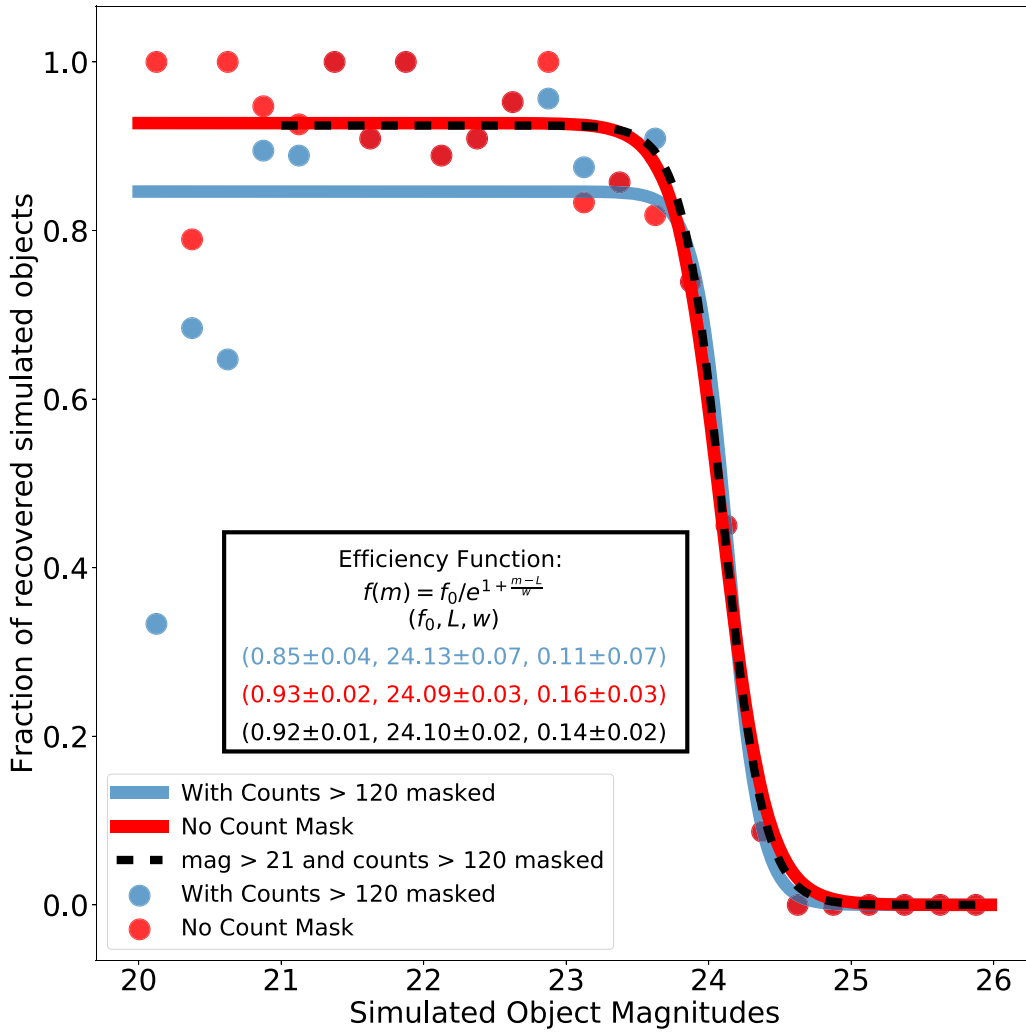


Figure 11. Comparison of efficiency curves run on the same simulated object set with and without the count masking described in Section 5.2. There is almost no effect on the depth of our recoveries.

fields, we calculate that the number of expected objects in this magnitude range to be less than 0.3. We find this to be an acceptable trade-off for the computational savings in this particular example. When going to difference imaging or with a different population of brighter expected objects we do not plan to include this extra masking step.

Another decision we made was to set our threshold for detection at a limit of 10σ instead of a lower, typical catalog level threshold of 5σ . This decision was motivated by the use of science images and the extra noise that would be present as this threshold was lowered. We looked at the increase of false positives as we go from 10σ to 5σ by rerunning the same field as Section 4.1 without any simulated objects and the time stamps randomly scrambled so that any detections were false positives. We show histograms comparing the false positives to true detections at the two thresholds in Figure 12. We also calculated the precision of our final results where precision is defined as $\frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}}$ and show the results in Figure 13. While we do detect objects to over a magnitude fainter ($g \sim 25.4$) with the lower likelihood threshold, we are overwhelmed by false positives at the fainter magnitudes and even at magnitudes at which we achieve high precision with a

higher threshold. At a threshold of 5σ we are below 50% precision at $g = 24.1$, while we are never below 90% precision when using a 10σ threshold. This degradation at a brighter magnitude in the 5σ results happens because bright false positives with fewer observations will appear at a higher likelihood than a dim object with more observations (e.g., a greater number of the observations were in a masked area or off the edge of the CCD). Due to this false positive performance when running on science images, we used the 10σ threshold in this work to show the effectiveness of the algorithm and code while also producing useful results. We discuss our future plans in the next section including what we will do to run the code at a lower detection threshold going forward.

5.3. Future Work

There is ongoing work to decrease the detection threshold for sources in order to extend the searches over longer timescales thus allowing us to find fainter and more distant objects. Work with longer baselines and larger stacks of images would also allow us to push this detection limit lower and confidently go beyond the limits of current population studies

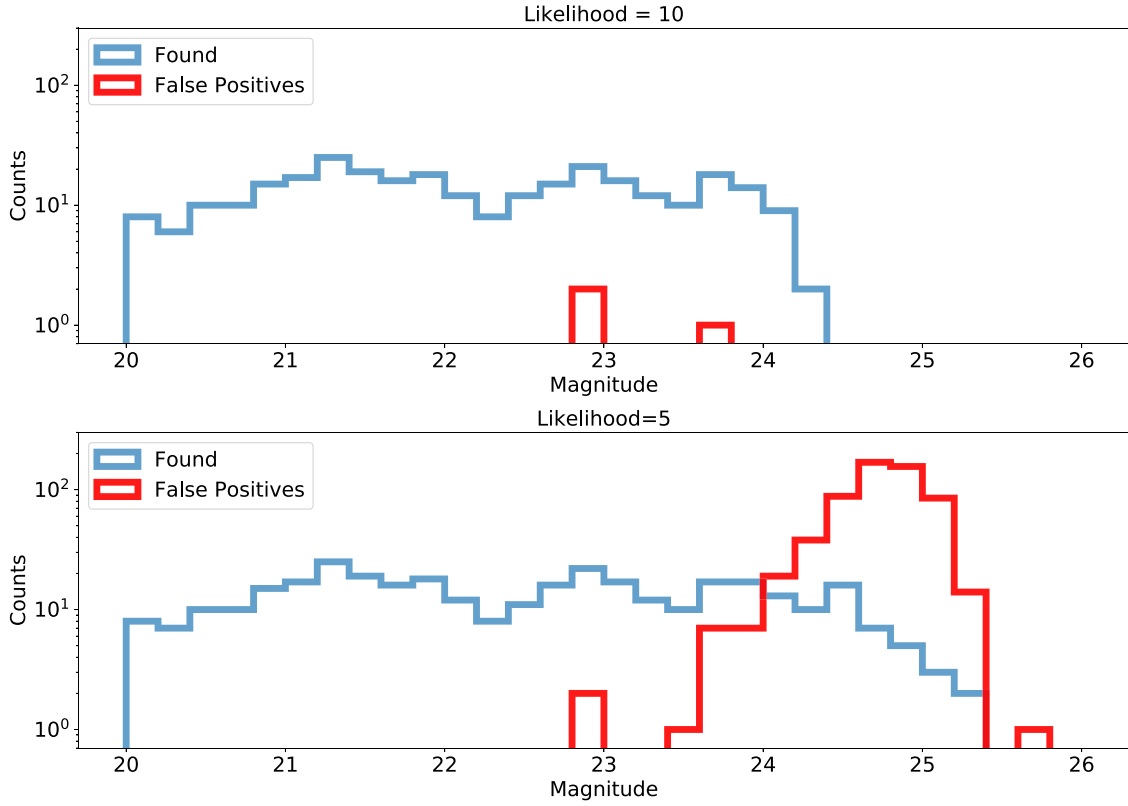


Figure 12. Comparing the false positives at detection thresholds of 10σ vs. 5σ in a field with simulated objects inserted. False positives overwhelm our filtering methods at the lower threshold when using science images.

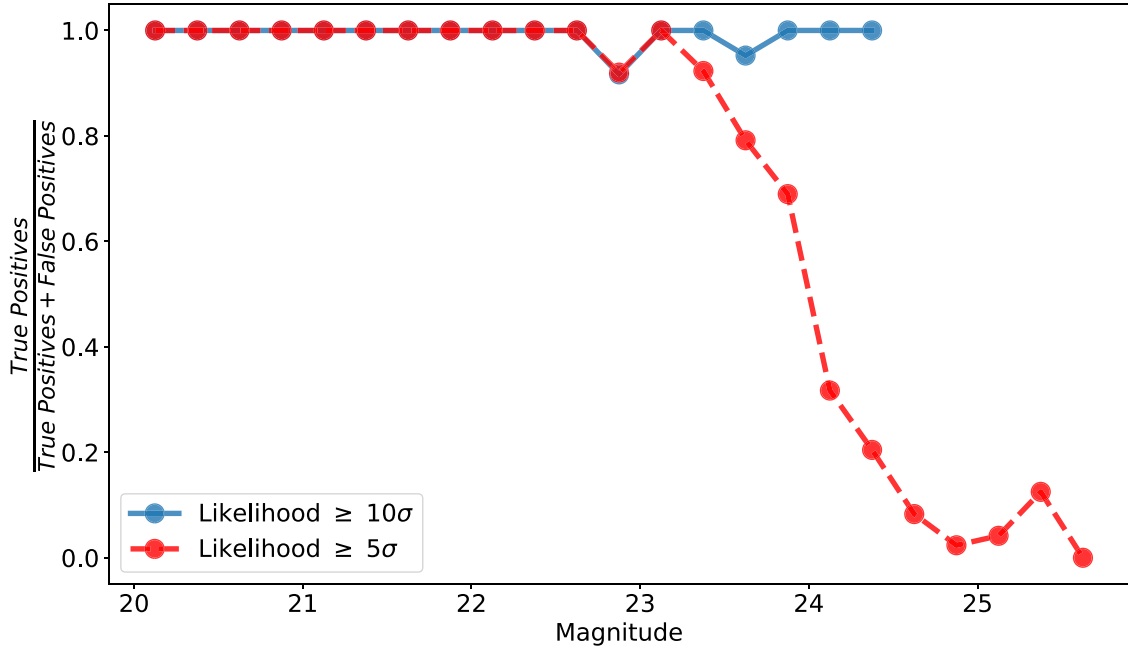


Figure 13. Comparing the precision at detection thresholds of 10σ vs. 5σ in a field with simulated objects inserted. We are very confident in our detections at 10σ with science images but would not be if we were to go to 5σ detections with the science images.

of KBOs. We plan to use the code with difference images in the future which we hope will reduce the time spent filtering and also remove many of the artifacts from science images that lead to false positives above 5σ and also pushed us to add in additional masking procedures in this work. We also plan to move to a deep learning based postage stamp classifier that we

hope will perform well at fainter magnitudes and plan to do a comparison versus our existing technique on the difference imaging search. We will also address working with nonlinear trajectories and methods to find objects that move from one chip to another during the survey period. Finally, we are also working on enhancements to the GPU algorithm that will allow

us to spend less time searching low likelihood trajectories thereby increasing search efficiency.

All of these improvements will help us explore deeper into the Kuiper belt and enhance our ability to use stacks of images from long baseline surveys such as Zwicky Transient Facility (ZTF) or LSST in the future.

6. Conclusion


In this paper we presented a new algorithm that uses the power of GPU processing to search for slow moving sources across a sequence of images. Our approach is capable of searching over 10^{10} candidate moving object trajectories in one minute. Applying these techniques to existing data we discovered 39 new KBOs and the recovered 6 KBOs already present in the MPC catalogs. Finally, we used the results of our search to compare to the Kuiper belt inclination distribution of Brown (2001) and the magnitude distribution of Fraser et al. (2008). We found both of the models to be consistent with observed results indicating that the recovered sample matches overall published characteristics of the KBO population. Combined with the high rates of detection efficiency recovered in tests, this indicates that our software provides a nearly complete recovery of an unbiased sample of moving objects down to the detection limit.

Our software, Kernel Based Moving Object Detection (KBMOD) is available to the public at <https://github.com/dirac-institute/kbmod>. This includes the GPU searching code as well as python analysis code we have used to process the search results. Development is continuing and can be followed at the Github repository hosting the code.

We would like to thank the anonymous referee for very helpful feedback that improved this paper. We also wish to thank Jim Bosch for valuable insights when we were starting this project. A.J.C. and J.B.K. acknowledge partial support from NSF awards AST-1409547 and OAC-1739419. A.J.C., J. B.K., and P.W. acknowledge support from the DIRAC Institute in the Department of Astronomy at the University of Washington. The DIRAC Institute is supported through generous gifts from the Charles and Lisa Simonyi Fund for Arts and Sciences, and the Washington Research Foundation.

Software: KBMOD (Whidden et al. 2018), LSST DM Stack (Jurić et al. 2015), astropy (Astropy Collaboration et al. 2013), scikit-image (van der Walt et al. 2014), numpy (Oliphant 2006), CUDA (Nickolls et al. 2008), scikit-learn (Pedregosa et al. 2012), pandas (McKinney 2010), matplotlib (Hunter 2007).

ORCID iDs

J. Bryce Kalmbach  <https://orcid.org/0000-0002-6825-5283>
 Andrew J. Connolly  <https://orcid.org/0000-0001-5576-8189>
 R. Lynne Jones  <https://orcid.org/0000-0001-5916-0031>
 Hayden Smotherman  <https://orcid.org/0000-0002-7895-4344>
 Colin Slater  <https://orcid.org/0000-0002-0558-0521>
 Andrew C. Becker  <https://orcid.org/0000-0001-6661-3043>

References

- Allen, R. L., Bernstein, G. M., & Malhotra, R. 2001, *ApJ*, **549**, L241
 Astropy Collaboration, Robitaille, T. P., Tollerud, E. J., et al. 2013, *A&A*, **558**, A33
 Bernstein, G., & Khushalani, B. 2000, *AJ*, **120**, 3323
 Bernstein, G. M., Trilling, D. E., Allen, R. L., et al. 2004, *AJ*, **128**, 1364
 Bosch, J. 2015, Algorithms for Detection and Coaddition, <https://github.com/lsst-dm/algorithm-docs/blob/master/2015-07-det%2Bcoadd-slides/slides.ipynb>
 Bosch, J., Armstrong, R., Bickerton, S., et al. 2018, *PASJ*, **70**, S5
 Brown, M. E. 2001, *AJ*, **121**, 2804
 Denneau, L., Jedicke, R., Grav, T., et al. 2013, *PASP*, **125**, 357
 Ester, M., Kriegel, H.-P., Sander, J., & Xu, X. 1996, in Proc. Second Int. Conf. Knowledge Discovery and Data Mining, KDD'96, ed. E. Simoudis, J. Han, & U. Fayyad (Palo Alto, CA: AAAI Press), 226
 Flaugher, B., Diehl, H. T., Honscheid, K., et al. 2015, *AJ*, **150**, 150
 Förster, F., Maureira, J. C., San Martín, J., et al. 2016, *ApJ*, **832**, 155
 Fraser, W. C., Kavelaars, J. J., Holman, M. J., et al. 2008, *Icar*, **195**, 827
 Gladman, B., & Kavelaars, J. J. 1997, *A&A*, **317**, L35
 Heinze, A. N., Metchev, S., & Trollo, J. 2015, *AJ*, **150**, 125
 Hunter, J. D. 2007, *CSE*, **9**, 90
 Johnston, L. A., & Krishnamurthy, V. 2002, *ITAES*, **38**, 228
 Jurić, M., Kantor, J., Lim, K., et al. 2015, arXiv:1512.07914
 Kaiser, N. 2004, Addition of Images with Varying Seeing, http://spider.ipac.caltech.edu/staff/fmasci/home/astro_refs/PanStars_Coadder.pdf
 Kalman, R. E. 1960, *J. Basic Eng.*, **82**, 35
 Kubica, J., Denneau, L., Grav, T., et al. 2007, *Icar*, **189**, 151
 Lang, D., Hogg, D. W., Jester, S., & Rix, H.-W. 2009, *AJ*, **137**, 4400
 McKinney, W. 2010, in Proc. 9th Python in Science Conf., ed. S. van der Walt & J. Millman (Austin, TX: Scipy), 51
 Nickolls, J., Buck, I., Garland, M., & Skadron, K. 2008, *Queue*, **6**, 40
 Oliphant, T. E. 2006, A Guide to NumPy (Spanish Fork, UT: Trelgol Publishing)
 Pedregosa, F., Varoquaux, G., Gramfort, A., et al. 2012, arXiv:1201.0490
 Reed, I. S., Gagliardi, R. M., & Stotts, L. B. 1988, *ITAES*, **24**, 327
 Rozovskii, B. L., & Petrov, A. 1999, *Proc. SPIE*, **3809**, 152
 Shao, M., Nemati, B., Zhai, C., et al. 2014, *ApJ*, **782**, 1
 Szalay, A. S., Connolly, A. J., & Szokoly, G. P. 1999, *AJ*, **117**, 68
 van der Walt, S., Schönberger, J. L., Nunez-Iglesias, J., et al. 2014, *PeerJ*, **2**, e453
 Whidden, P., Kalmbach, J. B., & Smotherman, H. 2018, Dirac-institute/kbmod: Paper Release, Zenodo, doi:10.5281/zenodo.1342298