google.org

RE<C: Multiscopic Photometry for Heliostat On-Target Spot Position Sensing

Overview

The high-performance receivers that convert heat to power for CSP power plants require careful temperature management. As described in our Pitch/Roll Heliostat Control System Design, we believed it necessary to place the light reflected by each heliostat on a specific section of the receiver to reduce thermal stresses on the receiver. To achieve a desired placement accuracy of 10 cm, our control system needed a precise way to determine the location of each heliostat's spot.

We extended the light spot sensing algorithms described by Kribus et al. in their 2004 paper. They relied on circumsolar radiation, a halo-like phenomenon where the brightness of the sky around the sun doesn't just drop off to nothing, but reduces gradually due to atmospheric light scattering. Cameras mounted around the target, each facing the heliostat field, capture circumsolar radiation. Kribus demonstrated that using the difference in brightness from camera images was a strong enough feedback signal to place a light spot near a target's center.

Our control system and experimental demonstration had two requirements beyond what the Kribus et al. paper described. The first requirement was that we wanted to place light spots at designated (x,y) locations on target, rather than at the center of the target. The second requirement was our desire for completely automated 7x24 operation. Our controller needed to automatically switch from the accelerometer-based coarse orientation sensor system to the precision sensing system, so it needed to know when the precision photometry signal was available for a heliostat light spot's location.

We designed a multiscopic photometry system that used four cameras situated around the target to triangulate the position of reflected light spots. From each set of images captured by the cameras, our software determined the brightness of the circumsolar radiation visible from each heliostat. We then used a map of brightness versus distance to convert this circumsolar

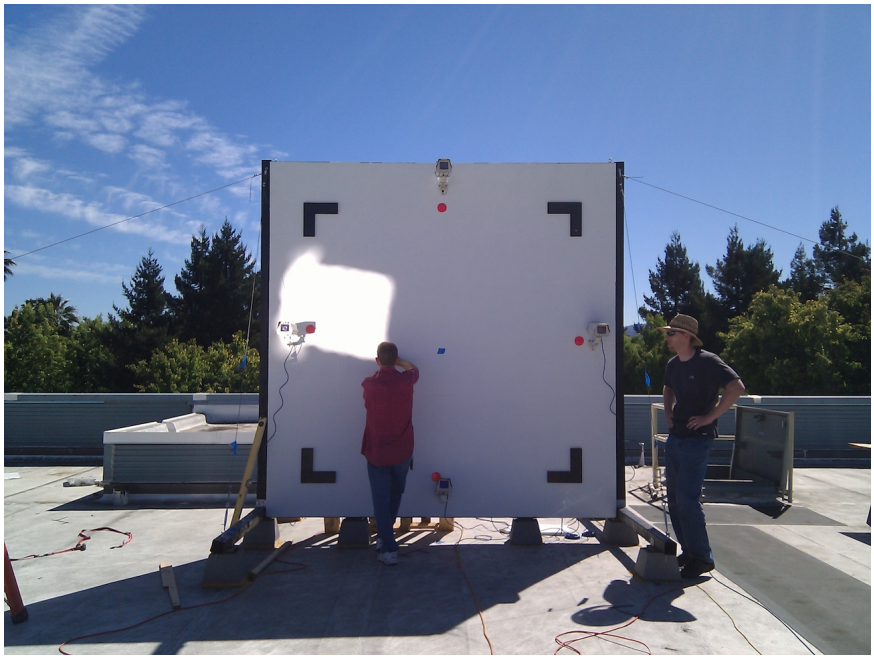
radiation measurement into an estimated distance between the heliostat's spot and the camera. For each heliostat, we combined the estimated distances and the known location of each camera to estimate the (x,y) location of the heliostat's spot on the receiver.

Our precision spot sensing system had a limited radius of operation, so our heliostat control system automatically switched back and forth from relying on the heliostat's accelerometer orientation sensor to relying on the on-target spot position sensing system for far higher on-target control. To do so, our multi-scopic system had to detect when a light spot came within its capture radius, and only then return a valid (x,y) position estimate.

This document details the photometry system we designed, experiments we conducted, the results we achieved, and our thoughts on worthwhile follow-up investigations.

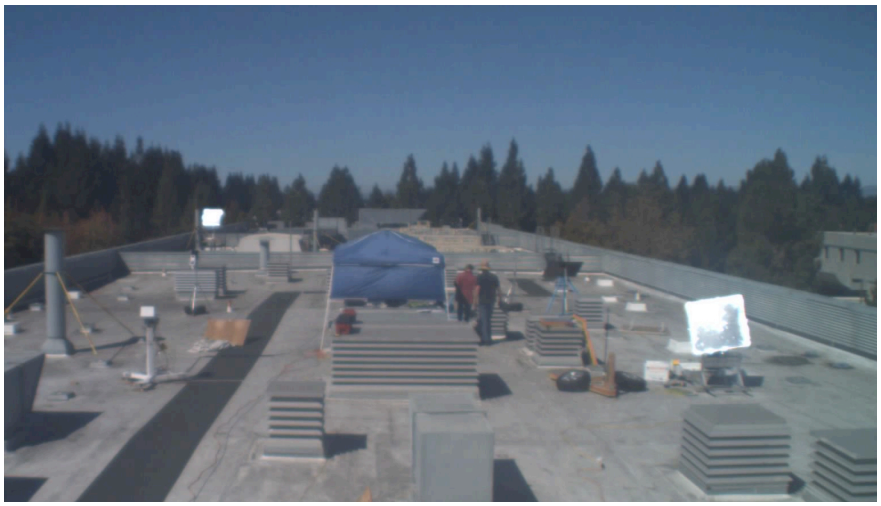
Capturing Circumsolar Radiation

To accurately capture circumsolar radiation, we positioned four cameras around the target aimed at the heliostat field. Each camera on the target could see every heliostat's mirror.



Our test target, illuminated by one heliostat. Note the four cameras next to the red spots (actually retroreflectors) along the edges of the target.

When a spot is near the camera, what the camera sees reflected in the mirror is the bright patch of sky somewhere near the sun.



View from the uppermost camera on our roof lab, with two heliostats visible and on target.

The intensity of the reflected circumsolar radiation falls off in a mostly monotonic manner within about ten degrees of the sun (see the circumsolar radiation graph below). Even if the sun is a bit beyond the radius of the cameras, a useful circumsolar signal is available.

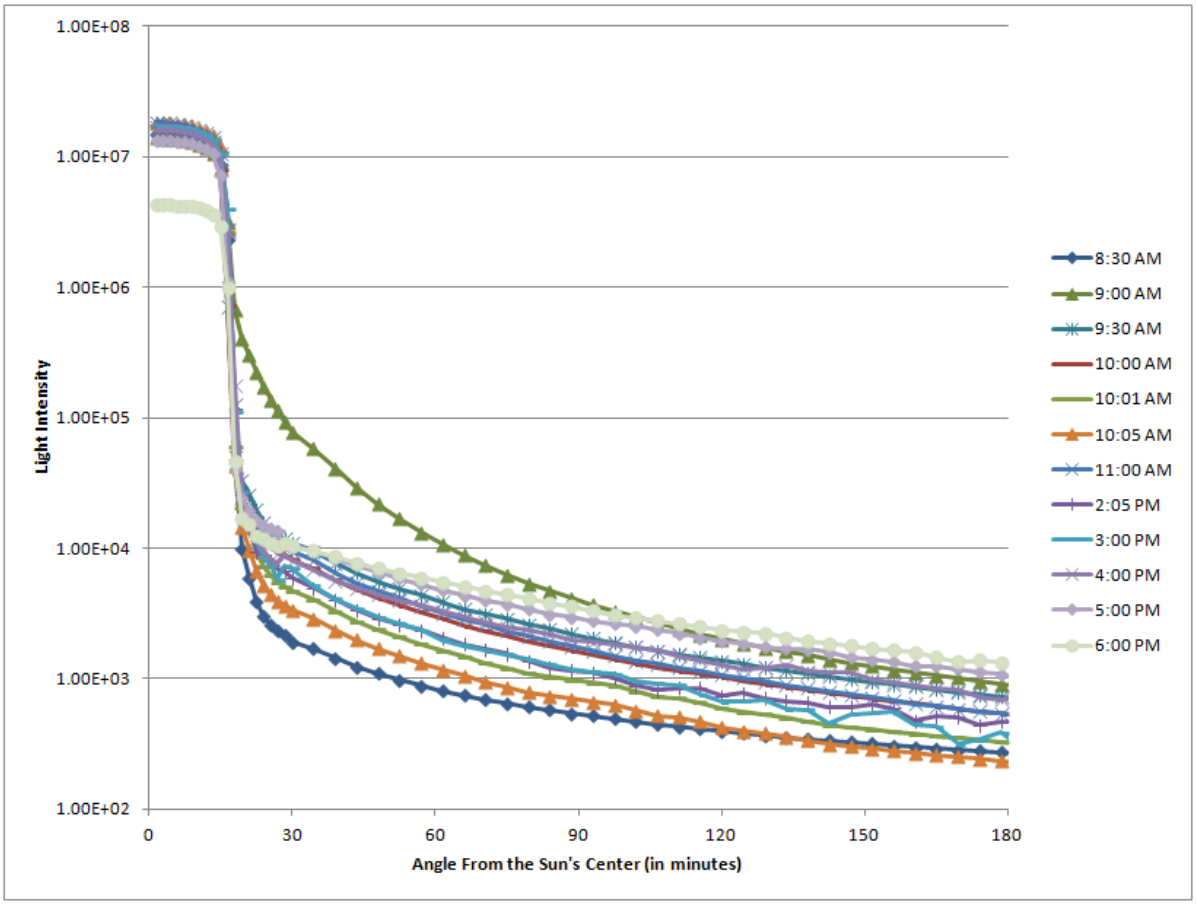
Mirror Identification

For each image captured by our field-facing cameras, we determined which pixels correspond to each heliostat. In our prototype, these were manually specified. See the conclusion for ideas on automating this process. Once the pixels that correspond to a heliostat were identified, the software measured their brightness. We used the mean brightness number for each heliostat, as seen by each camera, as input to our calculations.

Under certain conditions like the one shown in the picture above, the brightness of the pixels for one heliostat varied sufficiently to suggest a sunward direction. Our prototype did not take advantage of this phenomena.

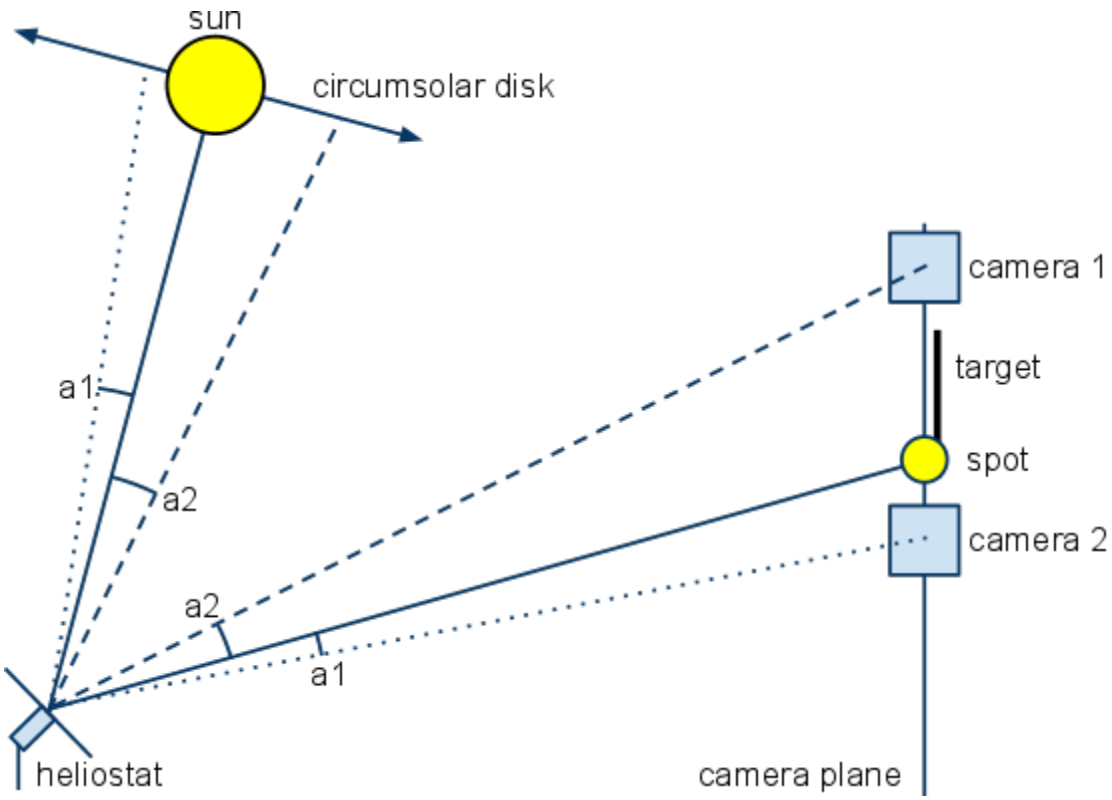
Estimating Spot Distance From Measured Brightness

Circumsolar radiation varies widely depending on multiple factors, including the angle from the sun, location, time of day, clouds, and haziness. The graph below shows a day of circumsolar radiation measurements published by the Lawrence Berkeley Laboratory. While the light intensity generally decreases the further from the sun, there is a lot of variation within a day. This means there is no simple formula, or even a universal single-valued function, to go from light intensity to an angle from the sun center measurement.



Circumsolar radiation measured during one day, data from NREL.

For our prototype, we used a manually compiled table that correlated measured intensity directly to on-target distance. Our prototype tests worked because we chose to operate under consistent sky conditions: the Northern California afternoon summer sky on clear days.



The intensity of light seen by each camera is a function of the spot's coordinates in the camera plane. Here, the squares are cameras and the small circle is the spot.

The following Matlab code shows the distance estimation approach we used for our prototype:

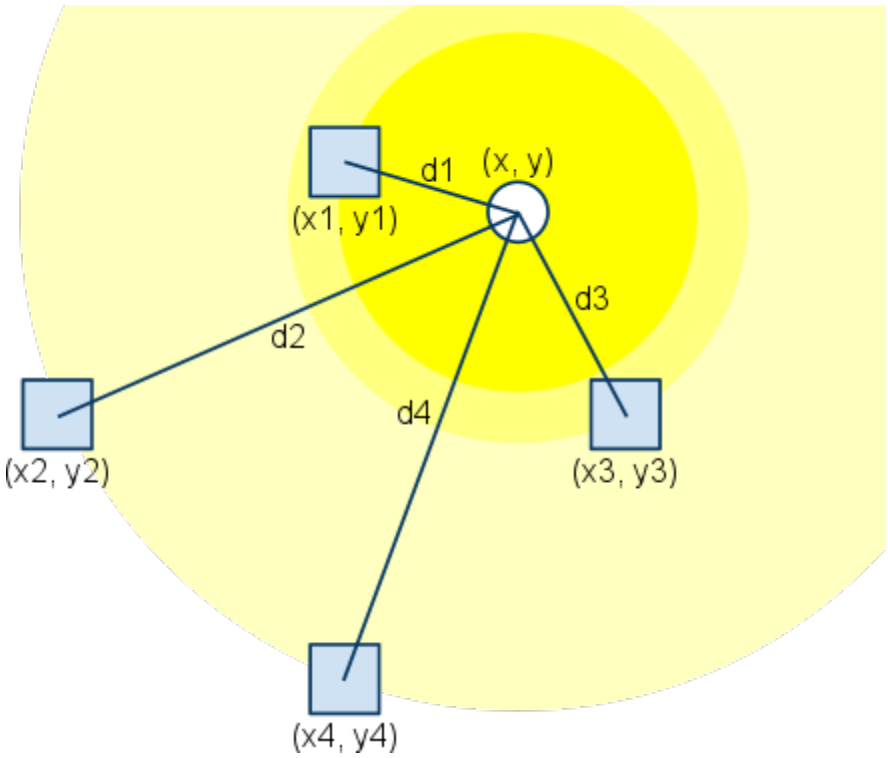
```
function [estimated_distance] = EstimateDistanceFromBrightness(brightness)
  % EstimateDistanceFromBrightness -- Estimates distances to the spot.
  %
  % EstimateDistanceFromBrightness(brightness)
  % Converts the brightness measures into estimated distances between
  % the camera that took the measures and the reflected light spot.  It
  % is specific to our setup; the conversion table was established by
  % manually moving the spot to a specific location and measuring the
  % brightness at that position.
  %
  % Arguments
  %
  % brightness: A vector of brightness measures.  It's a summation of the
  %             three red, green, and blue channels.  The maximum value is
  %             765.
  %
  % Return Values
  %
  % estimated_distance: The corresponding vector of distances.  If the
  %                     brightness is too low, the corresponding distance
  %                     will be NaN.

  % Manually measured table that maps a distance to a brightness.  This
  % table was specific to our camera settings and locations, and the sky
  % conditions found in clear Northern California skies.
  estimated_distance = interp1(...
```

```
        [765 723 631 423 352 294 248 215 198 173 167 149 143 134 116 70  65], ....
        [0.0 0.2 0.4 0.6 0.8 1.0 1.2 1.4 1.6 1.8 2.0 2.2 2.4 2.6 2.9 5.2 6.2], ...
        brightness);
end
```

Although this proved our concept, we only tested this during clear summer days in California. This is insufficient for a real-world system. In the future research section we list a few more ideas on how to solve this problem more generally.

Spot Location Estimation

At least three cameras have to report a good estimation of spot distance for a given heliostat, before we can estimate the heliostat's spot location via trilateration.



From the light intensities measured by the four cameras (the squares), we can estimate the distances (d1-d4) and compute the center of the spot (x, y).

Finding the center of the spot is similar to finding the epicenter of an earthquake. Measuring brightness instead of seismic waves, each camera measures brightness and correlates it with a radius. The point at which the radii meet is the center of the light spot.

Let's denote location of camera *i* in the target plane as a 2D vector $\boldsymbol{v}_i$, the distance between the spot center and camera *i* as d_i, and the spot center as $\boldsymbol{s}$. Assuming that all the measurements are accurate, we can write for each camera *i*:

$$(v_i - s)^2 = {d_i}^2$$

Let's expand the square in the left hand side, and pull all terms to the left:

$$v_1^2 - 2v_1 s + s^2 - d_1^2 = 0$$
$$v_2^2 - 2v_2 s + s^2 - d_2^2 = 0$$
$$v_3^2 - 2v_3 s + s^2 - d_3^2 = 0$$
$$\dots$$

Now let's subtract second equation from the first one to cancel out the s^2 term, and bring the terms without s to the right hand side:

$$2(v_2 - v_1)s = v_2^2 - v_1^2 + d_1^2 - d_2^2$$

This equation is linear in s. Now if we combine all pairs of equations in a similar way, we will get a system of linear equations, which can be solved with respect to the components of vector s:

$$\left\{2(v_j - v_i)s = v_j^2 - v_i^2 + d_i^2 - d_j^2,\ \forall\, i<j\right\}$$

Note that with three or more cameras, this system of equations is overdetermined, and in general can only be solved in the least squares sense. We found the least squares solution for circle intersection to work well in determining the spot location. The following Matlab code illustrates our least squares solution:

```
function [estimated_spot_center] = FindHeliostatSpotCenter(...
    camera_location, estimated_distance)
  % FindHeliostatSpotCenter -- Finds spot center based on distance.
  %
  % [estimated_spot_center] = FindHeliostatSpotCenter(camera_location, ...
  %     estimated_distance)
  % This function estimates the center of a heliostat spot from the known
  % camera locations and the estimated distance between each camera and the
  % center of the spot.
  %
  % Arguments
  %
  % camera_location: A matrix of camera locations, each row representing
  %                  the (x, y) coordinates of the camera in the camera
  %                  plane.
  % estimated_distance: For each camera specified in camera_location, the
  %                     estimated distance between the camera and the
  %                     center of the spot.  This value can be NaN.
  %
  % Return Values
  %
  % estimated_spot_center: The (x, y) coordinate in the camera plane of the
  %                        center of the spot.  If the arguments are
  %                        invalid or fewer than three distances are
  %                        provided, [NaN NaN] is returned.

  % Validate that the arguments are valid.
  if size(camera_location, 1) ~= size(estimated_distance, 1) || ...
```

```
     size(estimated_distance, 2) ~= 1 || ...
     size(camera_location, 2) ~= 2
    estimated_spot_center = [NaN NaN];
    return;
  end
  % Compute the least squares solution for circle intersection.  See writeup
  % at http://www.google.org/google_heliostat_multiscopic_photometry.pdf
  %
  % for every unordered pair (m, n) with m != n.  This is the linear system
  % A X = w, which we solve using least squares.
  A = [];
  w = [];
  for m = 1 : size(camera_location, 1)
    if ~isnan(estimated_distance(m))
      for n = 1 : m - 1
        if ~isnan(estimated_distance(n))
          A = [A; camera_location(n, :) - camera_location(m, :)];
          w = [w; (norm(camera_location(n, :))^2 ...
                   - norm(camera_location(m, :))^2 ...
                   + estimated_distance(m)^2 ...
                   - estimated_distance(n)^2) / 2];
        end
      end
    end
  end
  % Validate that we have at least three distances.  We may have fewer if
  % our caller could not convert all the measured brightnesses to a
  % distance.
  if size(A, 1) < 2
    estimated_spot_center = [NaN NaN];
  else
    estimated_spot_center = lscov(A, w);
  end
end

function Test()
  camera_location = [-1 0; 1 0; 0 1; 0 -1];

  % Test with four cameras reporting reasonable brightness:
  brighness_measured = [423; 207; 207; 677];
  estimated_distance = EstimateDistanceFromBrightness(brighness_measured)
  spot_center = FindHeliostatSpotCenter(camera_location, estimated_distance)
  assert(norm(spot_center - [-0.4681; -0.5356]) < 0.0001);


  % Test with two cameras reporting background brightness:
  brighness_measured = [423; 60; 56; 677];
  estimated_distance = EstimateDistanceFromBrightness(brighness_measured)
  spot_center = FindHeliostatSpotCenter(camera_location, estimated_distance)
  assert(isnan(spot_center(1)));
  assert(isnan(spot_center(2)));
end
```

Using a simulator, we verified that this algorithm performed well under various levels of distance estimation errors.

Experimental Results

Light spot on target

The equipment used in our experiments were mostly off the shelf components. We used:

- Four off-the-shelf Ethernet-connected Elphel cameras, located at the top center, left and right center, and bottom center of the target.
- Manually inserted neutral density filters to reduce incident light intensity on the cameras.

For our experimental setup, we:

- Manually adjusted camera sensitivity to reduce blooming while remaining sensitive to most of the circumsolar radiation.
- Measured and created a distance-intensity function.
- Identified at startup where each heliostat was within the cameras' fields of view.

Many of our observations were conducted using a heliostat at a slant distance of 60 metres from the target. At this range, one degree of arc subtends about one meter on the target. Despite our quick and imprecise setup, we observed tracking precision of better than 10 cm for our heliostats at a distance of 60 m when the spot was on the target and within the ring of cameras. Additionally, the direction to the spot was reported well enough within a radius two to three times larger than the ring of cameras that it allowed the control system to bring the spot onto the center target area using photometry alone.

During passing clouds, we observed that the photometry system provided enough signal to keep the reflected light spots on the target (although not necessarily at their commanded position) with quick recovery as soon as bright sun returned.

We did discover a problem when heliostats reflected sun directly onto a camera. Even with filters and adjusted camera settings, the reflected sun would overwhelm the imager, which recorded very large spot affecting many more pixels than the heliostat's mirror normally would.

An illustration of the blooming problem: two cameras' views of the same heliostat whose spot is directly on the right camera. Even with reduced exposure, the sensor of the right camera still saturates. The left camera was tuned to normal exposure to make the heliostat more visible.

When such blooming occurs, the intense spot produced by one heliostat hides the surrounding heliostats. We set our filters and camera aperture and exposure to minimize this problem, but there is no guarantee this will work for all field conditions. If the blooming phenomenon is not suppressed, the photometry system will erroneously think that all heliostats within the bloom are pointing directly at the camera. Since we only need three cameras to triangulate the light spot on the target, we can still determine the location of the light spot. However, additional code needs to be be written to detect and ignore data from a camera that's blinded in this way.

Future Research

Our initial results showed that we can estimate the light spot's position accurately. We proved that it was possible to achieve high-quality light control with our reduced-cost heliostat. These cost-cutting measures included using minimal on-heliostat sensors, lower cost cable drive

actuators, allowing some structural flex, and using low-precision installation on the field.

While testing the performance of our control system using this spot-position sensing system, we observed several issues that would need to be addressed to create a robust real-world system. We summarize these areas below should others wish to build on our research.

Automated Mirror Identification

For practical operation in the field, automatically determining which camera pixels correspond to which heliostat is a necessary feature. From the description of the field (location of the cameras and the heliostats) and knowing the optical characteristics of the camera, it's possible to establish the rough location of each heliostat in the image. An edge detection algorithm could provide a final level of adjustment.

If the mirror identification based on knowledge of the field is not sufficient, then one could potentially move a heliostat in a spiraling pattern, aiming to bring the reflected spot close to each camera. For each camera, the system would notice which pixels varied in brightness and would assign these to the heliostat being moved. Multiple heliostats could be tuned in parallel if we knew roughly the section of the image they appeared in.

Estimation of the Distance from Spot Center to Camera

Our multi-scopic photometry system performed well under blue skies but remained largely untested under cloudy or hazy conditions. Since the variability in circumsolar radiation cannot be estimated by a static model, we thought of different ways to measure it.

One possibility would be to measure the circumsolar radiation profile in real time using a camera pointing at the sky. After calibration, we could determine circumsolar flux measurements and use those instead of manual calibration to estimate angle offset from the sun.

Another potentially cheaper alternative would be to perform continuous re-calibration of the circumsolar radiation profile based on existing heliostat brightness measurements as seen by the very same cameras used for position determination. The circumsolar disk is the same for all heliostats on the field and it varies slowly during the day. The determined circumsolar profile can be built up over time using data from many observations.

Camera Survivability

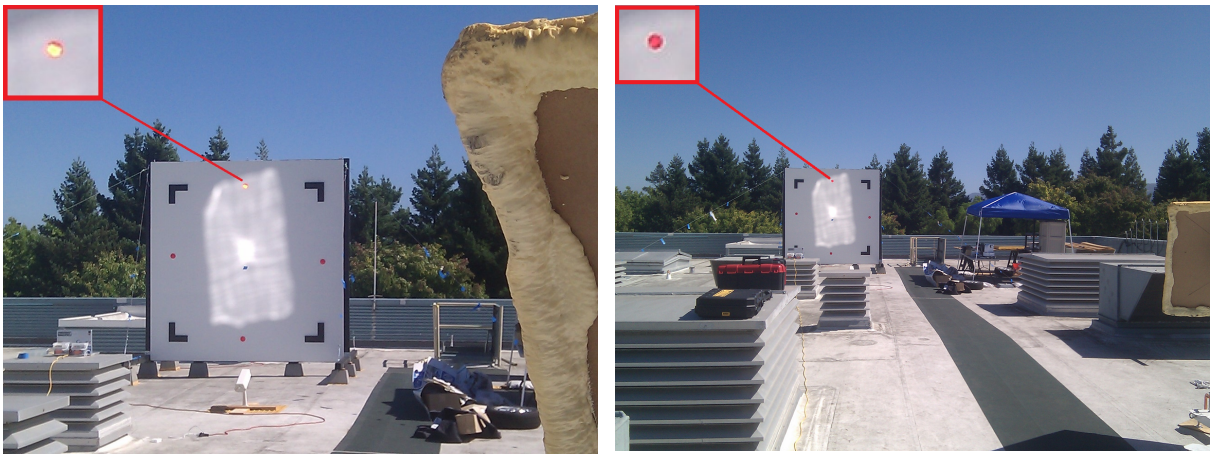
We had concerns that cameras could be destroyed (or images degraded due to blooming) if too many heliostats placed their reflected light spots on them. While careful positioning of the cameras could potentially reduce this risk, a powerful gust of wind could momentarily put a lot of flux on a camera. There are two main approaches that we thought might help, one involving severely limiting flux on the camera, and the other using retroreflectors at camera locations and doing sensing at each heliostat.

Concept #1: Small Aperture in Front of Camera

A special-purpose small aperture camera has the advantage of dramatically reducing the flux. A small aperture is set in a metal plate which acts as a thermal barrier to protect the imaging

camera behind it. Assuming sufficient residual dynamic range, blooming could be reduced as well.

Concept #2: On-target Retroreflectors with Heliostat-Based Sensing
Trihedral prism retro-reflectors could be used instead of cameras around the receiver target. Their reflected light signals could provide equivalent photometry information directly to each heliostat.



Two views of a target-mounted retro-reflector: the left view directly beside the heliostat that is reflecting the spot, while the right view is observed elsewhere.

The insets at the top left of the two images above highlight the reflection from the retroreflector located at the top center of the target. A useful property of retro-reflectors is that they send light back almost exactly to where it came from. The above pictures show that when viewed from the immediate vicinity of a heliostat, a reflector's light spot's illumination of a retroreflector is clearly noticeable, but it is not visible from other (heliostat) locations in the field.

If one were to place a rotating optical chopper in front of each retro-reflector, each rotating at a different frequency or with different timing, then one cheap photo sensor (e.g. IR detector, solar cell) on each heliostat could be demodulated to yield the brightness of each heliostat's reflected spot at the retroreflector locations. This information can be used as direct inputs into the multi-scopic photometry algorithm as discussed in this document. Glass retro-reflectors easily exceed 1 milliradian in accuracy, so blooming (inter-helistat interference) would likely not be an issue. Onboard heliostat sensors eliminate the need for automated mirror identification.

Conclusion

The photometry system was part of a systems approach to using lighter, cheaper heliostat structures imprecisely installed in the field. Though manually calibrated, our multi-scopic photometry prototype combined with our closed-loop control performed very well. Our ad-hoc testing showed that the spot position sensor was key to maintaining precise control: A heliostat could be moved, tilted or turned while tracking, and it would recover from such disturbances.

There remains important work to be done to improve the robustness of photometry sensing for real world conditions, such as effect of dirt on optical surfaces, resolving contrast in the face of large signal dynamic range, overexposure, etc.