Behavior Research Methods manuscript No.
(will be inserted by the editor)

psiTurk: A framework for running behavioral experiments online

First Author · Second Author

Received: date / Accepted: date

Abstract Insert your abstract here.

Keywords Online experiments · open science

1 Introduction (AC)

Online experiments are growing in popularity in the behavioral sciences. While they offer some exciting opportunities to researchers, they also pose a new set of challenges, both technical and experimental.

psiTurk is a framework of software and web services that addresses these challenges. It facilitates the creation, deployment, and sharing of web-based experiments on Amazon Mechanical Turk (AMT), which is currently the most common platform to run human experiments online. It solves a number of technical requirements including serving an experiment on the web, saving experimental data, and restricting the participant pool according to the experimenter's needs. It also provides an interface with AMT to create and test new projects as well as to manage and pay participants.

The principal goal of this framework is thus to handle common technical challenges so that researchers can focus on the development and dissemination of online experiments.

F. Author
first address
Tel.: +123-45-678910
Fax: +123-45-678910
E-mail: fauthor@example.com

S. Author
second address

2 Web-based behavioral research

It is easy to see why online experiments are an attractive option for behavioral scientists. Compared to traditional methods of experimentation in the laboratory they allow researchers to collect large data sets in a fraction of the time and at much lower cost. At the same time, they also open up a completely new set of web-based tools that can be used to design the look and flow of an experiment and that far exceed the capabilities of traditional experiment software. They also cater to a wide range of needs from simple surveys to more complicated interactive interfaces, or even multi-player games.

Still, there exists a range of hurdles that researchers need to tackle when running experiments online, some of which might altogether bar some scientists from adopting online experimentation. To find out more the specific needs and challenges we conducted a survey in early 2014 of behavioral researchers regarding online experimentation.

2.1 Survey of researchers' needs (AR)

We collected responses from 201 researchers in psychology (58%), linguistics (16%), marketing (7%), neuroscience (6%), and economics (4%), most of whom had some experience collecting behavioral data online in their labs (85%).

These researchers showed a clear interest in online data collection. Nearly all listed fast data collection (98%) and large sample sizes (93%) as benefits of online data collection, with most interested in the potential for lower costs (75%) and a more diverse population (60%) as well. Nearly 40% stated that they treat pa-

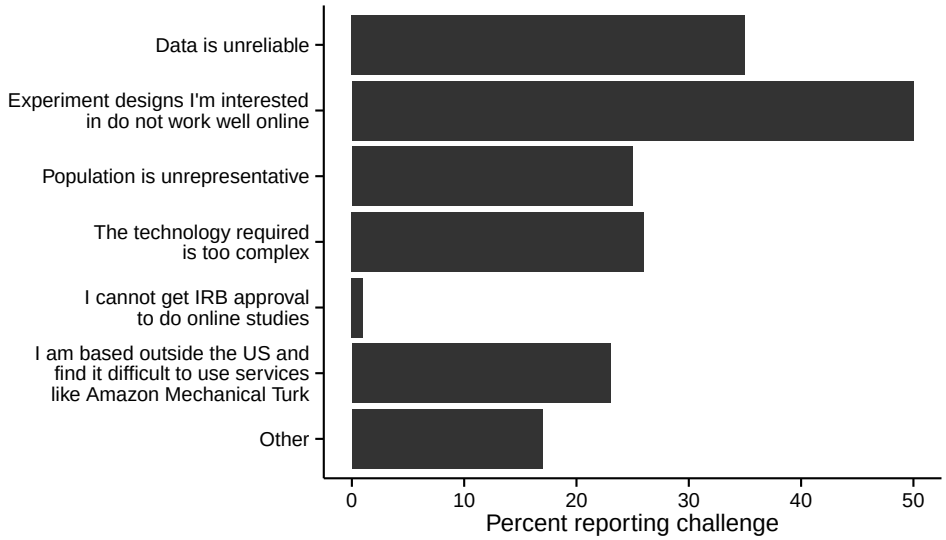

Fig. 1 Challenges faced by 201 researchers who completed our survey on collecting behavioral data online.

pers reporting data collected online identically to that collected in a lab.

Respondents also felt that online data collected presented a set of unique challenges, as shown in Figure 1. Some felt that data collected online was unreliable (35%) or the population unrepresentative (25%). Many felt that experiment designs they were interested in did not work well online (50%), or that web technology is too complex (25%), and researchers based outside the US reported difficulty using services like Amazon Mechanical Turk (23%).

The majority of respondents (56%) listed Qualtrics, a service for conducting online surveys, as the tool they were currently most likely to use, but most (79%) expressed interest in being able to run full experiments including multiple, trials, fixation crosses, etc. The vast majority (94%) reported interest in new tools that simplified online data collection.

2.2 Amazon Mechanical Turk

Amazon Mechanical Turk (AMT) is an online platform that lets you post a wide variety of tasks to a large pool of participants. People who complete tasks, so called *HITs* (Human Intelligence Tasks), are called *workers*, people who post tasks are called *requesters*. Instead of spending weeks to run experiments in the lab, AMT lets you collect data of a large number of workers within a couple of hours. Workers get paid a fixed amount for each HIT which is determined by the requester.

From a worker's perspective, the AMT website offers an overview of all available HITs with a title and brief description, as well as the basic payment for completion. A worker can then click on a HIT to see its full *advertisement*, which is an embedded website that is either hosted by Amazon or the requester, and decide to accept the HIT. The task itself can then either be completed on a requester's own hosted service, or on the AMT website itself, if the requester uses an Amazon template. At the end of the task, a worker needs to submit the HIT in order to receive compensation. Some HITs also offer bonus payments that are determined by the requester based on a worker's performance.

From a requester's perspective, there are two main avenues for creating HITs. For very basic tasks like surveys and open text fields, AMT offers a range of built-in templates. For more complex tasks, like most behavioral experiments in psychology, a requester has to provide their own hosted service which runs the experiment, saves the data, and interfaces with AMT at the beginning and end of the task.

2.3 What problems does psiTurk solve? (AC)

To address the needs and concerned that survey participants mentioned, the psiTurk platform offers three key services.

2.3.1 Interface to AMT

psiTurk provides an easy to use interactive command line tool that can be used to interact with AMT without ever having to use the AMT website. The command line tool can be used to create and manage HITs, both

live and in the AMT sandbox (AMT's testing environments for HITs before they go live), and to pay workers for participating. Besides the basic payment that every worker receives for completing a HIT it also has the capability to give automated bonus payments to participants based on a function designed by the experimenter. This feature is particularly helpful since the AMT website requires bonuses to be entered by hand for each participant individually.

2.3.2 Technical complexity and security

psiTurk reduces some of the main technical challenges that researchers face by providing a web server and basic data base to host an experiment and store participants' data. It also solves one very specific problem by providing a secure ad server that can be used to serve the advertisements that workers see on the AMT website. Without a secure server ads would not display on AMT and thus the ad server servers as a bridge between AMT and a researcher's local computer. Someone may have to check if this is true :) While providing a basic server and database, psiTurk also allows researchers to add custom server routes or alternative databases that can be tailored to their specific needs.

2.3.3 Recruiting, filtering, and instructing participants

Finally, psiTurk also has a range of features of particular interest to behavioral researchers running controlled experiments. For example, it offers basic capabilities for assigning workers to different experimental conditions and counterbalancing features of the experiment. It also provides a rage of templates for common experiment building blocks (e.g., consent form, debriefing, instructions). In addition, it allows experimenters to filter workers for example to exclude certain devices, platforms, or browsers, or to prevent the same worker to take a current or previous HIT repeatedly.

2.4 Remaining challenges

Some of the concerns raised by survey respondents are inherent to online data collection and therefore psiTurk does not address them. It is worth pointing out however that there exists some work on data reliability - one of the greatest worries that was found in the questionnaire - that successfully replicates a wide range of classic cognitive psychology findings using AMT data (cite crump et al). Interestingly, this study also found that increasing worker payment had no effect on reliability, suggesting that even at low payment levels data quality was high.

On the other hand, there clearly exist experimental protocols that simply are not amenable to online experimentation. Experiments that require very fine grained temporal control over stimulus presentation, for example, may be unsuited because even modern browsers will not be able to reliably display content fast enough if the presentation time becomes too short (also see crump et al). Similarly, any experiment that requires control over a participant's screen size, resolution, or distance from the screen will be problematic due to the uncontrolled nature of participants' environment. One advantage of psiTurk is that it automatically collects data on worker's interaction with the experiment window, that is, if and when the window was resized and if and when the user switched tabs or windows. This might help to at least get some grasp on a worker's level of distraction while completing the task.

We will continue to discuss these challenges at the end of the paper.

3 Overview of the psiTurk system

The structure of the psiTurk system is shown in Figure X.

AMT
psiTurk experiment server (local computer)
Ad server
Experiment exchange

In this section we highlight important functions of each component. Complete documentation and tutorials are available at http://psiturk.org/docs .

3.1 Configuration and project structure

After creating an Amazon Web Services account, a researcher must obtain credentials (a *key ID* and a *secret key*) that are saved in a global configuration file `.psiturkconfig`. When psiTurk is run for the first time this file is automatically created in the user's home directory, but the user's AWS credentials must be added in order for psiTurk to interact with AWS.

In addition, need a pair of psiTurk keys to allow posting of ads on the ad server, and these are also placed in `.psiturkconfig`. This global configuration is used whenever psiTurk is run, allowing multiple experiments (each residing in its own directory somewhere on the user's system) to rely on the same AWS and psiTurk accounts, or to share other common configuration options as desired.

A new psiturk project can be initialized using:

```
$ psiturk-setup-example
```

which creates a new directory containing an example experiment. The project directory contains the following:

- `config.txt`: Contains settings for this particular experiment
- `custom.py`:
- `participants.db`: This sqlite database file is created by default in the experiment file
- `server.log`: A log file containing any messages from the experiment server (not from the actual experiment code!)
- `static` directory: Experiment files, including Javascript, CSS, and images
- `templates` directory: HTML files associated with experiment

Setting up a new experiment thus entails 1) editing the settings in `config.txt`, followed by 2) modifying the contents of the `static` and `templates` directories to program the experiment. The remaining files are only necessary in order to debug server errors, add new server-side functionality (`custom.py`), or to access saved data (`participants.db`). To run your experiment in a web browser you need to have at least some basic web programming skills (especially using HTML, CSS, and JavaScript). Once you mastered the basics, you can take advantage of the vast number of libraries and tools that can help you to build sharp and sophisticated experiments with the support of a large community of users.

3.2 Command line interface: Managing HITs and serving the experiment (AR)

PsiTurk runs as a command line interface (CLI) within a standard terminal window. There are several advantages to designing psiTurk as a simple CLI beyond its efficiency of use. This design makes the user-interface code clear and easy to read and write, allowing newcomers to quickly understand and contribute to the open-source project. Integrating a new feature into the interface is as simple as describing the syntax and functioning of a new command. The CLI also ensures that psiTurk is easy to interact with not just on a laptop or desktop but also on a remote server or in the cloud, where users may have terminal-only access.

Entering `psiturk` at the command line in any directory containing a psiTurk project launches the psiTurk CLI. The psiTurk CLI features a colorized prompt that provides important information at a glance, including whether the server is running, the current mode ("live" or "sandbox", and the number of HITs currently running on AMT:

```
[psiTurk server:off mode:sdbx #HITs:0]$
```

In some examples below, this prompt is truncated as `[psiTurk]$` to indicate commands that occur within the psiTurk CLI, while commands that are entered on the command line are preceded simply by `$`.

3.2.1 Managing HITs

Commands are organized into groups based on their function, following a general "`command subcommand arguments`" format. For example, one can create a HIT by typing

```
[psiTurk]$ hit create <# assignments> <$
    amount> <duration>
```

What happens after you create a HIT?

During the development of an experiment, the current code can be tested using:

```
[psiTurk]$ debug
```

which will open a new browser window in which the current experiment can be tested.

Can restart the server with `server restart` or open the server log with `server log`. Tab completion makes it quick to type commands, and the `help` command can be used to show details on the usage of any command. The straightforward and consistent syntax of the CLI allows users to perform a wide variety of tasks– from creating HITs and paying workers, to launching Amazon Web Services database instances, to opening an experiment in a browser for debugging–that would otherwise be spread across a number of websites and programs. Easily switch between AMT's sandbox and live site.

In most cases, a user of the psiTurk CLI will never have to log into the MTurk website except to add money to their MTurk requester account.

3.2.2 Serving an experiment

The experiment server is started using

```
[psiTurk]$ server on
```

This opens the port specified in the project configuration file. The experiment server will then respond to incoming requests, assuming that the port is publicly accessible. This requires a static IP to prevent the experiment's URL from changing. Users without a static

IP address can use a dynamic DNS service to forward requests to their dynamic IP. If the system running psiTurk is behind a router, the router must be configured to forward requests on the same port.

3.3 The Ad server

PsiTurk helps an experimenter in both hosting the ad and delivering the experiment. First, PsiTurk provides a cloud service at `psiturk.org` for serving ads to workers. This service serves the ads with appropriate SSL certificates, something which can be difficult on many server configurations.[1] Second, the PsiTurk web framework allows you to easily build and host your own experiment, running on your server or laptop and saving workers' data either to your own database or to `psiturk.org`. This gives experimenters complete flexibility in terms of content while also offering a suite of tools to make experiment development easy.

SSL hosted Ad Server removes need to deal with complex web security issues (https, data mining on workers) Participants recruited via Mechanical Turk first interact with your task via ads. Ads are simply the digital version of hanging a poster or flyer around your university building in order to recruit participants. Technically, ads are snippets of HTML code that describe what your task is about and what you're offering for compensation. As a result, they are the front line for any subject recruitment online. It's easy to overlook the importance of a good ad, and making that ad visible to as many participants as possible.

3.4 Javascript library *psiTurk.js*

The javascript library *psiTurk.js* enables interaction with the server from the client-side (javascript) experiment code. The goal of the library is to handle the most common functionality of psiTurk-based web experiments, without imposing any additional requirements on the structure or design of the experiments themselves. Researchers can draw on a vast array of javascript libraries to design their experiment, while using psiTurk.js to save data and notify the server of changes in a participant's status.

Figure X shows javascript code from an experiment script (*task.js*). *psiTurk.js* currently performs the following functions:

[1] SSL certification has recently become a requirement for hosting content within an `iframe` on `mturk.com`.

3.4.1 Tracking a participant's progress in an experiment

psiTurk records changes in a participant's status as they move through an experiment. Some of these status changes are automatic, e.g., when a participant is assigned to a condition or if they quit an experiment early. Two additional status changes are initiated by the client-side experiment code using psiTurk.js functions. First, since experiments typically begin with an instructional phase, calling the function `psiturk.finishInstructions()` upon completion of this phase will save the participant's status as having begun the main experiment. If the participant attempts to quit the experiment after that point, they will receive a warning that they will not be able to restart the experiment and may forgo payment. Second, successful completion of the experiment is signaled by calling `psiturk.completeHIT()`, which closes the experiment and redirects the participant to the AMT page to submit their HIT.

3.4.2 Saving experimental data

Experiment code can save data in two formats. `psiturk.recordTrialData` takes any array of values as input and appends it to a list of "trial" data. This data structure is meant for sequential data that may be collected over the course of multiple trials or blocks, where each line corresponds to a new measurement. However, the format of this data is defined by the experimenter and is saved in the database as a single JSON object.

In contrast, `psiturk.recordUnstructuredData` is used to record (key, value) pairs, where the key is uniquely defined within the experiment. This format is useful for survey questions or other one-time measurements, e.g., (*Age*: 24).

Importantly, both functions above simply record the data in the appropriate format on the client-side. When the experimenter wishes to save the data to the server they call `psiturk.saveData`.

3.4.3 Automatic recording of browser interaction

One general shortcoming of web experiments is greater uncertainty about a participant's testing environment and engagement with the experiment. Unlike lab computers where undesired behaviors can be prevented, a person is always able to close a web browser, switch to different applications, or change other aspects of their experience. However, standard methods exist for recording many aspects of a user's interaction with a

web browser, and this data can be useful for 1) tracking how an experiment was actually displayed, and 2) the level of a participant's engagement.

For example, although it is possible to set the initial size of a browser window, a web participant can change the dimensions of the window, potentially obscuring or altering how the experiment is displayed. *psiTurk.js* automatically records these changes in the size of the window. Similarly, the experiment can choose to switch focus away from the experiment window (e.g., to another browser window or a different application). *psiTurk.js* automatically records every time that the experiment window loses and gains the participant's focus. This "event data" is automatically recorded and saved to the database whenever `psiturk.saveData` is invoked.

psiTurk.js can also preload html pages and images. Finally, it has a basic structure for presenting instructional pages.

3.5 Experiment exchange

A significant advantage of web-based experiments is the potential for low-friction replication and extension. The *experiment exchange* (EE, found at http://psiturk.org/ee) facilitates the sharing of experiments that have been built to run using psiTurk, acting as an "app store" for psiTurk-compatible designs. Once an experiment has been completed, a researcher can submit the following information to register their experiment on the exchange:

1. A Github repository containing the experiment code
2. Metadata about the experiment, including a title, description, and keywords
3. The DOI for any paper describing the results of the experiment (if any)
4. The version of psiTurk that was used to run the experiment

Metadata associated with an experiment allows other researchers to discover it through the psiTurk website. A publicly available experiment can then be downloaded using the following command on the command line,

```
$ psiturk-install <experiment-specific-id>
```

with the experiment-specific identifier found on the exchange page. If psiTurk is already configured on the user's system, the downloaded experiment will run without further changes, and can then act as a starting point for direct replication or extensions. Modified versions can then be re-run using the same population via AMT, thereby minimizing the potential for experimenter-specific biases.

Other initiatives (e.g., the Open Science Framework, https://osf.io) also aim to improve the transparency, reproducibility, and efficiency of research through centralized services, but are less focused on the specific technology involved in running online experiments. In contrast, the psiTurk experiment exchange links together experiments that can be run within a common framework. As a result, existing experiments can serve as examples to help new researchers learn to code for the web; they shorten the development time necessary (especially for popular experimental paradigms); and the exchange facilitates communication between researchers with similar interests.

4 Limitations and future directions (TMG)

Some challenges that were noted by the survey are inherent to online data collection

Restriction to AMT, can be an unrepresentative participant pool.

No Windows

Counterbalancing (integrate PlanOut (Facebook's counterbalancing tool)) (DH)

Optimal Experimental Design/ Design of Experiments tools

Better multi-person/multi-session exp (websockets; doug?)?

Automate catch trials? How far did the worker get? Do we count them in the N?

Limiting/throttling number of people doing the experiment at once

Video support