# Needs for Research

Computer literacy is an important issue. Traditional methods for gaining literacy n computers are taking courses and reading books. This is good for acquiring general computer literacy. There are also many specific computer knowledge. Users can visit the official site of the software vendor or consult the official manual or read the built-in help. But the amount of information is limited. That’s why there’s a market for computer books. Reading the books is hard. It is restricted to the book on the user’s bookshelf. Books can be expensive.

Fortunately, there are tons of such knowhow knowledge on the web.

That’s why people look at the Internet. People no longer need go to bookstores. People gain knowledge from the Web. Some knowledge is general and can be applied over and over. The easy to learn and remember. Some is transient and disposable, only done it once. The goal is to follow the steps accurately.

However, search engine has the largest coverage. It is too general. It’s not suitable for computer knowledge. There is no specialized system for accessing and searching this knowledge base. People are aware of this problem. So they built specialized search engines built for special domains, such as news to show times and images, products to show prices and images, restaurants to show locations and ratings. These were mined from unstructured heterogeneous data.

There are specific needs for each stage of the knowledge life cycle.

**Need to collect and index computer how-to resources on the Web.** Articles containing computer knowhow are scattered all over the Web. But there has not been any systematic effort to collect and index them. In order to build a specialized search engine for these articles, we need to be able to discover and identify all these different types of documents. This can be challenging because these articles take on various forms. They can be about various operating systems, Windows, Mac OS, Linux. They can be called guides, manuals, walkthroughs, tutorials, or how-to. They can be written for novice and experts. Some have only text. Some have screenshot images. Some are videos. The algorithm needs to recognize all of them while at the same time being able to avoid including articles about software but not related to knowhow such as product review and catalogs.

**Need for an indexing scheme optimized for computer how-to resources.** Schemes traditionally adopted to index Web articles are based on keywords. Articles are broken down into keywords. An inverted index is constructed for efficient retrieval, where each word is a key and stored under each key is a list of articles containing the word. However, this scheme is not optimal for computer knowhow articles. It has been observed that users can follow articles faster when the articles contain screenshots to illustrate each step compared to text-only articles [cite]. It is conceivable users may prefer articles with screenshots. Since these articles are indexed only by the keywords they contain, there is no way to search them by visual contents. Moreover, for efficient retrieval, we need a way to break images down into *visual* keywords as well so that we can add them to the inverted index. Thus, there is a need for an optimized indexing scheme that not only considers both textual and visual content but also lends itself to the construction of an inverted index for efficient retrieval.

**Need for an easier input method for specifying query terms.** Current input method for specifying queries is difficult. It is hard to come up with the right keywords. To identify the context, there are too many things need to be specified. Users need to indicate the operating system, the name of the application, the name of the window, and the topic, what the users want to know. The result is a long list of keywords, which can be ambiguous. The context and topic are indistinguishable from the search terms. From usability point of view, it is time consuming to type all the keywords.

**Need for a ranking scheme.** Current ranking schemes either based on text content or on image contents. However, simply ranking articles by images may result in articles at the top containing no useful text. Simply ranking articles by text may result in informative articles but on the wrong computer application.

**Need for a scheme to display excerpts to help users judge relevancy quicker.** Current presentation scheme tends to show except from the page by extracting excerpts and highlight the occurrences of the search term. This allows users to know how the search terms are in use. However, it is still difficult to quickly judge whether an article is relevant. The result may contain all the keywords, but without actually following the link to visit and read the page, the user may not really know whether the page is really relevant to the particular application window.

**Need for a more effective, convenient, interactive way to follow knowhow articles.** After the users found an article relevant to the task at hand, it is still challenging to actually follow the article. The users may already be at a step. It is hard to know where in the article the users can read about his current step. It is difficult to search within the reference for the part relevant to the current step. The users need to switch back and forth between the application and the reference. This is very inconvenient. It is even worse if the reference material is a screencast. The video player needs to be paused and played periodically. Seeking within the video is not possible.

**Need for a cheaper and accessible way to create contextually technical references.** Contextual help is useful. But its creation requires access to the API, which can be expensive. Thus, many references are created. But they are created independent of the applications, external the application. It is hard to link the page to the application.

# Goals and Objectives

The overarching goal of the proposed project is to address the needs identified in all stages of the knowledge life cycle as described above.

Develop an algorithm for detecting articles containing computer knowhow.

# Proposed Methods

We propose a multi-modal approach. We use images and text.

## Creation

Allow content providers to attach arbitrary information to programs by images.

Establishing visual links.

Submit an url to the system. And it downloads all the image and index the page.

## Aggregation

There are many existing knowhow knowledge on the Web. We want to aggregate them into a searchable index. The brute-force method would be to systematically crawl major tutorial websites. For this research, we can use a short cut method. Bootstrap method. Use image search engine, take words from a corpus such as a book. Mix in words related to software name. Keep those with screenshots. Train a visual detector for screenshot images. What properties define tutorial knowledge articles?

It is necessary to filter useless pages that are not related to computer knowledge. There are text based methods. Also an image based method. Train a classifier for detecting pages that are computer knowledge.

We used three methods to collect screenshot images to populate our database. Currently, our prototype system contains more than 150k images in its index.

First, we submitted computer-related keywords to Bing Images to collect screenshot images of interactive programs. To increase the likelihood of obtaining the desired images, we sampled keywords from title bars of the dialog windows of various computer programs. Some examples of these keywords are properties, preferences, option, settings, wizard, custom, installation, network, sound, keyboard ... etc. We turned on the filter feature to keep only illustrations and graphics, rejecting obviously non-screenshot images such as images of faces and natural scenes. Using this method, we collected approximately 100k images.

Second, we used TinEye, a reverse image search engine that can take an image as the query and return a list of URLs to nearly identical copies of the image found on the Web; it is designed primarily for copyright infringement detection. We manually captured screenshot images of more than 300 interactive windows of popular programs across three of the most popular OS platforms (XP, Vista, and Mac OS). These images were submitted to TinEye to obtain about 5,000 images.

Third, we collected a library of 102 electronic books of popular software programs. We extracted all the image figures embedded in the electronic file (i.e., PDF documents). This method produced about 50k images.

Each method has its own pros and cons. While Bing Image Search provides the best variety of images, many of them are not visually relevant to any program at all. TinEye is able to provide visually relevant images, but these images are ranked only by visual relevancy; the page containing the highest ranked image may not necessarily contain any useful information. Computer books are professionally edited and thus contain the highest quality text; however, they cover a relatively limited range of applications and their content isn't as current as compared to the Web. By using all of these methods, we hope to create a rich repository of technical information that is both visually and textually relevant to, and accessible by, general computer users.

## Indexing

We propose to index knowledge by text and images. Specialized, multi-modal indexing scheme. Compared to keyword only and text only. Develop a scheme that uses images to create context and evaluate against a sizable dataset and show statistical significance more relevant results than start-of-the-art methods, when evaluated by human users. First page result. Above the fold result. Recall and precision. Use inverted index for fast retrieval. Offline processing.

## Ranking

query dependent. rank results by both visual and textual relevance. Identify many features. Learn ranking, using RankSVM. Improve visual search. Provide faceted search function.

## Querying

allow users to capture sceenshot as query. Allow users to type keywords as query. Use Java to provide cross-platform applicability. Allow users to take multiple screenshots as query.

## Previewing

show image excerpt. Show screenshot in context to let user know the context, what words are before and after that.

## Consuming

monitor the entire screen, matching the screen to the images in a tutorial article. Automatically scroll the page to that image. Allow users to search for content by image. Browser function called Find by image.