## 5.4 Lessons Learned

Each instructor gathered or developed videos as best fit their course. The right-most column of Table 1 gives an estimate of instructional time dedicated to creating and/or curating videos across the different courses. Even when videos are curated rather than created, the process of curating videos for a whole course takes a significant amount of time.

We wanted the students to benefit from learning course concepts from more than one perspective. This was particularly important in the Web Applications course, where students had a wide range of expertise from those who had never seen HTML to some who had taken courses in high school or even had some job experience. Providing a range of videos let students find resources that matched their current experience levels and expanded upon what we could cover during the semester.

Students utilized the video resources differently than a lecture, particularly for programming topics. They commented that they were able to follow along with the code on their own, and pause, rewind, and replay as needed. Presenting such examples inside a classroom, with students following along on their own computers, is much more difficult. One student said: *"The good thing about videos is you can pause/play as you please and rewind/fastforward when necessary,…"*

The use of videos from various authors demonstrated the variety of resources that exist on the Internet to help teach and provide examples of code and programming concepts. The diversity in resources promotes self-exploration to students, which they demonstrated later in their homework. Many referred back to the video tutorials while completing their homework, something that is not possible if the lecture is only available live in class. One student posted: *"As I am finishing up my final website, I am going back over the video tutorials for last minute help and ideas. They are much easier to understand at the end of the semester!"*

However, a disadvantage of curated videos was that they did not always present a topic at a conceptual level, nor were they tailored to the common misunderstandings or challenges that students in a particular class have. Students often described situations in which the author of the video didn't present information clearly. *"I thought these videos were very unhelpful. The guy just seems to be putting code on the screen and keeps saying "we will get to that later" and just continues to build his website."*

To alleviate some of these issues, we provided a brief high level conceptual explanation in class the week prior to the topic. However, some students reported this was not sufficient and they would have appreciated more conceptual information within the videos themselves. We are considering the addition of our own short conceptual lecture videos to address this issue.

Another drawback of online instruction is that watching videos is a solitary activity. We are trialing opportunities for students to comment, critique, and reflect on online videos in a more social way, through the Video Collaboratory [14]. The Video Collaboratory is an online system that allows groups of students to privately and asynchronously discuss videos through text and sketch annotations. For example, in our current HCI course, students are required to watch the videos inside the Collaboratory, and annotate them weekly. Members of the group can view each other's annotations and add to the discussion. We plan to extend the use of the Collaboratory to more courses as we gain experience in its use. We believe that the 'shared watching experience' in small groups of 5-10 students creates a more intimate and less intimidating forum for discussion around educational videos.

# 6. IN-CLASS ACTIVITIES

In addition to creating or collecting online videos, one of the other major challenges for flipped classroom instruction is creating class activities that teach course concepts and skills through active learning. For our programming courses, labs are a critical active learning component. But, the real challenge is to find active learning activities to replace the time that would traditionally be used for lecturing. We have developed two main types of in-class activities: group problem solving and flexible quiz activities.

## 6.1 Pair Programming Labs

Similar to many programming courses, the Web Applications and Media Computation courses include weekly pair programming lab sessions. Typically, students were asked to edit and extend existing code. In later parts of the course students had to code from scratch. The tasks covered specific skills and language constructs each week. Lab activities were carefully structured to enable completion within the time allocated, assuming the student had prepared. We have tested many strategies for pairing students in the lab, including gender pairing and randomly changing the pair each week or every other week. The purpose of switching partners frequently is to help the students get to know a larger number of students in their class and to make sure that students don't play the same role with the same person each week.

The use of pair programming was successful, and seemed to encourage students to work both with their partner, and over time with other students, to answer questions and learn new techniques. Alternating the pairs was also important, as the skill and knowledge level of students was quite varied.

## 6.2 Group Problem Solving Activities

For the non-programming courses, and on the non-lab day for the programming-based courses, the problem solving activity is a highly structured activity that is timed. Typically, students work in small groups to encourage conversation and peer instruction. In the HCI and Rapid Prototyping classes, these activities generally involved students practicing design and evaluation skills. For example, in the HCI class, when teaching needfinding, one or two students in a group were asked to develop a persona and role play a typical user, while the others in the group prepared interview questions and conducted an interview. In Rapid Prototyping, student groups utilized different techniques to create prototypes at different levels of fidelity. In Web Applications, activities included finding existing web pages to demonstrate weekly topics, drawing conceptual diagrams and writing pseudo-code. In Media Computation students often solved problems by writing pseudo-code or doing other paper problem solving activities.

In all courses, we emphasize collaboration and critiquing. These activities are then built upon later when students complete their assignments, which test their application of the learned skills without teacher supervision. In the HCI and Rapid Prototyping courses, students can re-use and extend some of their in-class activities as part of their assignments.

## 6.2 Flexible Quiz Activities

In flipped classrooms, instructors often use quizzes to incentivize students to watch videos and come to class prepared. We also use quizzes as learning activities, especially in the programming courses. This idea is based on the finding that retrieving information from memory improves long term retention [13]. The quiz generally comprised multiple choice questions about textbook and video content, or about code fragments. In contrast to the problem solving activity, the quiz tests and reinforces conceptual knowledge where the problem solving activity provides scaffolding for programming skills or HCI methods.

Throughout the four courses, different techniques were used to make the quiz an activity that encouraged peer learning and the discovery and clarification of misconceptions. We developed several strategies for student interaction around quizzes:

1. Students complete the quiz labeled only with their id number. Quizzes are redistributed to others for peer grading. The instructor directs the peer grading by discussing why particular answers are right or wrong for each question.
2. Students complete the quiz on their own, and then are asked to compare and discuss their answers with the students sitting near them until all students sitting near each other have the same answers. Each student's quiz is graded separately.
3. Students form groups and each group completes a quiz, again, ensuring that the answers are the consensus of the group. All students in a group get the same grade.
4. Students complete the quiz using clickers and can talk to each other before answering. The answers are aggregated on the screen in front of the room, and instructors re-poll as needed. Various game-like methods are used to create competition.
5. Student groups create a quiz based on their annotated videos. Each group takes another group's quiz and the group that created the quiz grades the answers. In this way each group creates a quiz, takes a quiz, and grades a quiz.

In all variations on the way the quiz was administered, the quiz provided the focus for the peer learning and the discovery of misconceptions. When misconceptions were uncovered, the instructors conducted impromptu mini-lectures. Thus, the quiz activities could often take up a significant portion of a class.

## 6.3 Lessons Learned

All of the in-class activities are designed to be collaborative in some way. We promote peer instruction for the following reasons: the social aspect of peer instruction helps students get to know other students and making friends is important for program retention; when students are socially relaxed they are likely to learn more; some students have an easier time learning from peers than from professors; learning from peers demonstrates that there are many sources of knowledge; learning from peers exposes students to different perspectives on issues; and, teaching peers helps students clarify and solidify their own understanding of course concepts.

In programming labs, we use pair programming with frequent partner switches throughout the semester. In our workshop classes we facilitate student collaborations in a variety of ways. In HCI there is a high-stakes group project and teams typically sit together and work together on in-class activities. In the Web Applications and Rapid Prototyping courses, ad hoc groups or pairs are formed for in-class activities. This works well in upper year classes where students are typically comfortable forming teams and students have friends that they want to work with.

In the Media Computation course, we created lightweight teams consisting of 5 students per team. These teams sit together in assigned seating during the peer instruction workshops each week. These teams are called 'lightweight' because there is no direct link between team activities and an individual's final grade, though the team works together all semester. We find this works well for freshmen courses where students don't know many people.

We found that emphasizing skill development during class activities enabled us to create more open-ended homework assignments that encouraged creativity. For example, in the Web Applications course, students created a website to promote a business or hobby. Many of the websites created were more professional, with greater content, than assignments from prior (non-flipped) years.

# 7. STUDENT FEEDBACK

The standard student evaluation forms administered at the end of the semester typically start with statements about the instructor. For example "My instructor displays a clear understanding of the course topics." "My instructor has an effective style of presentation." These create an impression that the quality of the course depends on how much the instructor knows rather than on how much the student learned. While it may be appropriate to evaluate the quality of the instructor, flipped classrooms require a survey that primes students to think about the quality of their learning experience. We administered a survey to gather student perceptions about the flipped classroom approach to provide formative evaluation and immediate feedback from students.

| Survey Item | Mean | SD |
|---|---|---|
| I felt I was more able to learn at my own pace (compared to previous courses). | 5.06 | 1.41 |
| I was more able to recognize when I didn't understand something (compared to previous courses). | 5.00 | 1.34 |
| I looked at extra information beyond the provided material (compared to previous courses). | 4.54 | 1.60 |
| I felt that I learned more during class time (compared to previous courses). | 4.89 | 1.59 |
| I felt that I learned more outside of class time (compared to previous courses). | 4.81 | 1.54 |
| How helpful were the online videos for learning? | 5.17 | 1.48 |
| Learning from other students is just as good as learning from the teacher. | 4.76 | 1.47 |
| Other students helped me to understand the right answers while we were discussing clicker questions. | 5.04 | 1.17 |
| I helped other students to understand the right answers while we were discussing clicker questions. | 4.97 | 1.20 |
| Explaining things to other students helped me to understand them better. | 5.35 | 1.09 |
| How mentally demanding was the course? | 4.55 | 1.24 |
| How successful were you in accomplishing what you were asked to do in this course? | 4.35 | 1.72 |
| How hard did you have to work to accomplish your level of performance in this course? | 4.95 | 1.36 |

**Table 3: Post course student survey average responses.**

The post-course survey was conducted across four courses with responses from a total of 213 students. While each survey contained course-specific items, the common part of the survey consisted of 13 Likert-type items with a 7-point rating scale, with